



Une formation
Alphorm.com

Formation Python pour *les Pentesteurs*



Hamza KONDAH



Une formation
Alphorm.com

Plan

Python pour la sécurité
Plan de la formation
Public concerné
Prérequis



Une formation
Alphorm.com

Python pour la sécurité

Librairies

Prototyping rapide (POC)

Outils disponibles

Communauté

Syntaxe



Une formation
Alphorm.com

Plan de la formation

Introduction

1. Le Web et Python
2. Développement d'exploits et Python
3. Forensics et cryptographie en Python

BONUS : Analyse de malware et Python

Conclusion



Une formation
Alphorm.com

Public concerné

Responsables de la sécurité des SI
Auditeurs en sécurité



Une formation
Alphorm.com

Prérequis

Avoir des bases en sécurité des SI
Connaître le fonctionnement des réseaux
Comprendre le fonctionnement du
protocole HTTP
Avoir des bases théoriques en cryptographie
[Python pour les pentesteurs : Partie 1](#)



Introduction aux architectures Web



Hamza KONDAH



Plan

Architecture Web

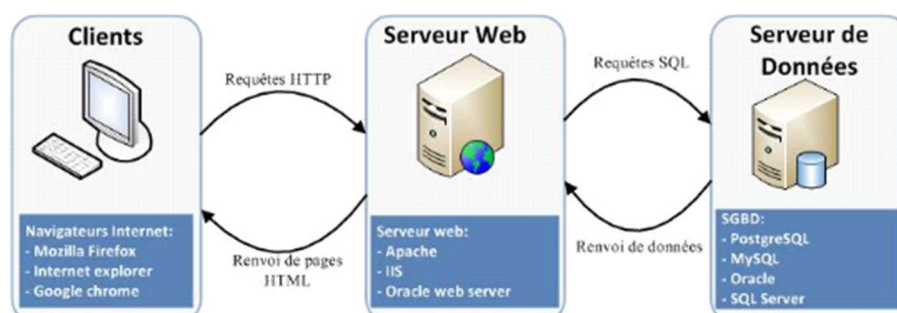
Éléments intéressants

Lab : Environnement Web

Une formation
Alphorm.com



Architecture Web



Une formation
Alphorm.com



Une formation
Alphorm.com

Éléments intéressants

Balises

Identification des champs

Headers

Problématique : Documents malformés

Non respect des
standards



Une formation
Alphorm.com

Lab : Environnement Web

Apache

HTTP SERVER



Merci



Extraire et analyser le contenu



Hamza KONDAH



Une formation
Alphorm.com

Plan

Introduction

Données Web

Parsing de données

Lab : Extraction et Analyse



Une formation
Alphorm.com

Introduction

Urllib et Urllib2

Analyse basique et avancée

Les outils les plus performants sont
en Python 😊



Une formation
Alphorm.com

Données Web

HTML, XHTML, JSON, XML

Nécessité de traitement des données

Gestion des données mais aussi
création de requêtes personnalisées

PS : La compréhension des
mécanismes d'attaques est primordiale



Une formation
Alphorm.com

Données Web

« *Parser un fichier est l'action
d'exploiter un fichier sous une forme
brute afin d'en tirer les informations
utiles* »



Parsing de données

LXML, BeautifulSoup, HTMLParser ...

Problématique : manque de
conformité avec les standards HTML

Le traitement avancé nécessite le
passage via des astuces

Une formation
Alphorm.com

Merci



Une formation
Alphorm.com

BeautifulSoup



Hamza KONDAH



Une formation
Alphorm.com

Plan

Définition

Lab : BeautifulSoup



Définition

Excellent parseur

Facile à exploiter

LXML + html5lib → Meilleur Parsing

Gestion excellente des encodages

La meilleure librairie sur le marché 😊

Une formation
Alphorm.com



Lab : BeautifulSoup

```
print soup.title           # <title>The Dormouse's story
print soup.title.name      # u'title'
print soup.title.string    # u'The Dormouse's story'
print soup.title.parent.name # u'head'

print soup.p               # <p class="title"><b>The Dor
print soup.p['class']      # u'title'
print soup.a               # <a class="sister" href="htt

print soup.find(id="link3") # <a class="sister" href="htt
print soup.find_all('a')    # [<a class="sister" href="htt
                           # <a class="sister" href="htt
                           # <a class="sister" href="htt
```

Une formation
Alphorm.com



Exercice

Mettre en place un Crawler

Mettre en place un Screen Scraper

Mettre en place un Whois

Une formation
Alphorm.com

Merci



Une formation
Alphorm.com

La Librairie Mekanize



Hamza KONDAH



Une formation
Alphorm.com

Plan

Introduction

Lab : Mekanize



Une formation
Alphorm.com

Introduction

Simulation de navigateur

Extrêmement puissant

Facilement manipulable

Parfait pour le test de vulnérabilité

Compréhension des scénarios de navigation nécessaire préalablement



Une formation
Alphorm.com

Lab : Mekanize

```
1 import mechanize
2 import time
3
4 class Transaction(object):
5     def __init__(self):
6         self.custom_timers = {}
7         self.base_url = 'http://reddit.tracelytics.com'
8
9     def run(self):
10        br = mechanize.Browser()
11        br.set_handle_robots(False)
12
13        start_timer = time.time()
14        resp = br.open(self.base_url + '/r/all/')
15        resp.read()
16        latency = time.time() - start_timer
17        self.custom_timers['All Items'] = latency
18        assert (resp.code == 200), 'Bad HTTP Response'
19
20 if __name__ == '__main__':
21     trans = Transaction()
22     trans.run()
23     print trans.custom_timers
```



Exercices

Proxy avec Mechanize (Récupération
des liens sur la page)

Vérification de vulnérabilités : SQLi et
XSS

Une formation
Alphorm
com

Merci



Une formation
Alphorm.com

Analyse de fichiers XML



Hamza KONDAH



Une formation
Alphorm.com

Plan

Structure de fichiers XML

Lab : XML et Services Web



Une formation
Alphorm.com

Structure de fichier XML

```
<?xml version="1.0" encoding="UTF-8" ?>
- <humain>
- <corps>
  <tete>petite tete</tete>
  - <bras>
    <bras_1>bras_1</bras_1>
    <bras_2>bras_2</bras_2>
  </bras>
  - <jambes>
    <jambe_1>jambe_1</jambe_1>
    <jambe_2>jambe_2</jambe_2>
  </jambes>
</corps>
- <esprit>
  - <lob_droit>
    <equilibre>l'equilibre</equilibre>
  </lob_droit>
  - <lob_gauche>
    <la_parole>la parole</la_parole>
    <des_sentiments>les_sentiments</les_sentiments>
  </lob_gauche>
  <pensee>la pensee</pensee>
</esprit>
</humain>
```



Une formation
Alphorm.com

Lab : XML et Services Web

```
#!/usr/bin/env python
#coding:utf-8
from xml.dom.minidom import Document
import xml.dom.ext
doc=Document()
noeud_racine=doc.createElement("Sommaire")
doc.appendChild(noeud_racine)
comment=doc.createComment("Ceci est un exemple")
noeud_racine.appendChild(comment)
titre=doc.createElement("titre")
noeud_racine.appendChild(titre)
contenu=doc.createTextNode("Programmation Python")
titre.appendChild(contenu)
element=doc.createElement("chapitre")
element.setAttribute("numero","1")
noeud_racine.appendChild(element)
titre=doc.createElement("titre")
element.appendChild(titre)
contenu=doc.createTextNode("Introduction")
titre.appendChild(contenu)
date=doc.createElement("date")
date.setAttribute("jour","14")
```

Merci



L'OWASP



Hamza KONDAH



Une formation
Alphorm.com

Plan

Introduction

Lab : L'OWASP



Une formation
Alphorm.com

Introduction

Open Web Application Security Project

Communauté (active) travaillant sur la
sécurité des applications Web

Libre et ouverte à tous

Recommandations de sécurisation Web

Méthodes et outils de référence

Plusieurs projets



Lab : L'OWASP



Une formation
Alphorm.com

Merci



Une formation
Alphorm.com

Les failles XSS



Hamza KONDAH



Une formation
Alphorm.com

Plan

Exploitation

Cross site Scripting

Lab : Cross site Scripting



Une formation
Alphorm.com

Exploitation

La réussite de l'exploitation des vulnérabilités nécessite une phase de scanning effectuée minutieusement

Le scanning peut nous renvoyer des faux positifs
Il est nécessaire d'effectuer des tests post scanning à la main pour s'assurer de la véracité de la vulnérabilité et d'en découvrir d'autres



Une formation
Alphorm.com

Exploitation

PS : Pas toutes les vulnérabilités sont découvertes grâce aux scanners
Exploitation faille par faille
Niveau très avancé
Balayage





Une formation
Alphorm.com

Cross Site Scripting

XSS

Injecter du contenu dans une page
Javascript

Simple ? Pas intéressant ? -> la
réponse est NON



Une formation
Alphorm.com

XSS

La brise qui cache la tempête
Redirection, exécution, frame, vol
d'informations, exploitations
avancées, etc





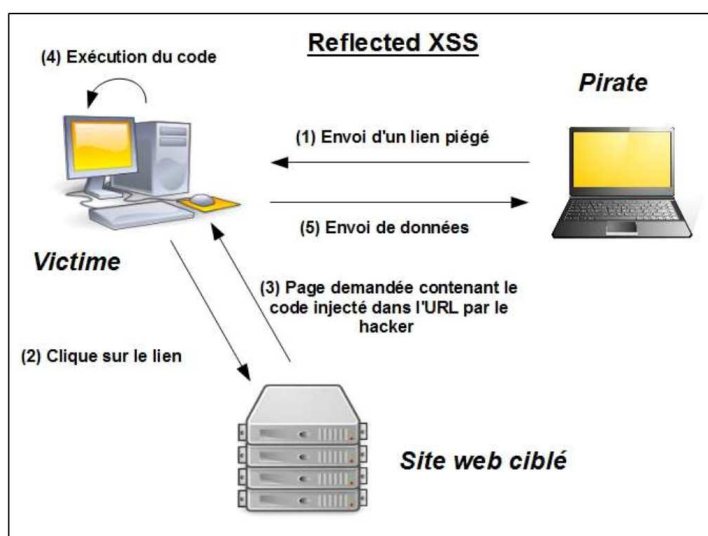
XSS

Une XSS non persistante
Très présent
Input non vérifiée
Injection de code
Manipulation de données
Ingénierie sociale

Une formation
Alphorm.com



XSS



Une formation
Alphorm.com



XSS

Persistante

Danger !

Attaques puissantes

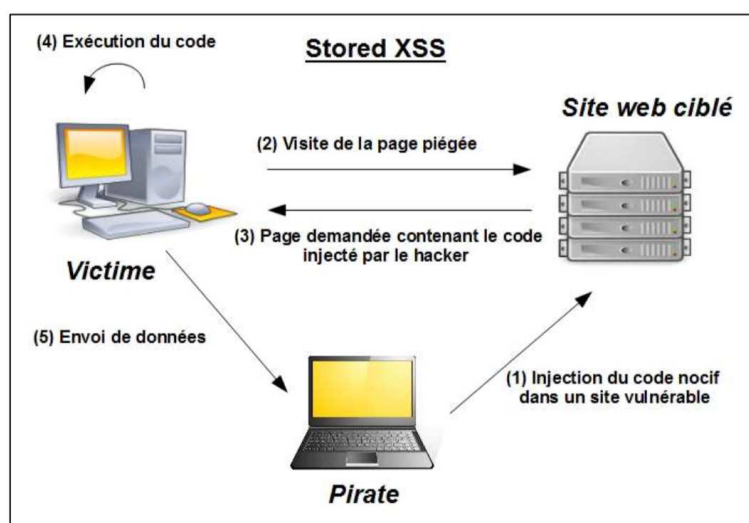
Bases de données, fichiers ...

Exemple : Forums

Une formation
Alphorm.com



XSS



Une formation
Alphorm.com

Merci



Shodan API



Hamza KONDAH



Une formation
Alphorm.com

Plan

Introduction

Lab : Shodan API



Une formation
Alphorm.com

Introduction

Moteur de recherche

Périphériques connectés (IPv4 et IPv6)

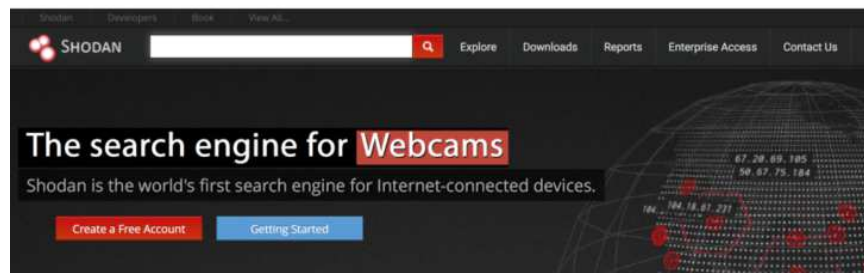
Recherche de machines spécifiques

Filtres customisés

API de développement



Lab : Shodan API



Une formation
Alphorm.com

Merci



Une formation
Alphorm.com

Introduction sur le développement d'exploits



Hamza KONDAH



Une formation
Alphorm.com

Plan

Introduction

Outils de recherche

Pour bien suivre 😊

Objectifs du module

Lab :Introduction



Une formation
Alphorm.com

Introduction

Platforms	Exploitation Techniques
Windows xp – SP1 , SP2 ...	Simple Buffer Overflows
Windows Vista – SP ...	SEH , SafeSEH
Windows 7 – SP ...	NX,DEP
Windows Server 2003,2008,...	ASLR
Linux	Stack Cookies
Mac OSX	...



Une formation
Alphorm.com

Outils de recherches

Ollydbg

Metasploit Framework

IDA PRO

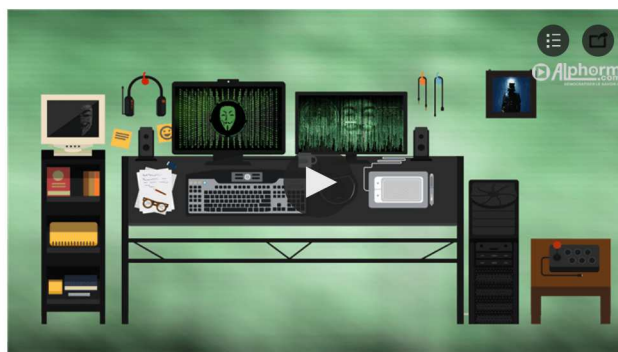
WinDBG

Et une tonne d'autres 😊



Pour bien suivre ☺

Formation Hacking et Sécurité : Acquérir les fondamentaux
Apprendre les techniques pratiques de base utilisées par un Hacker



Une formation
Alphorm.com



Objectifs du module

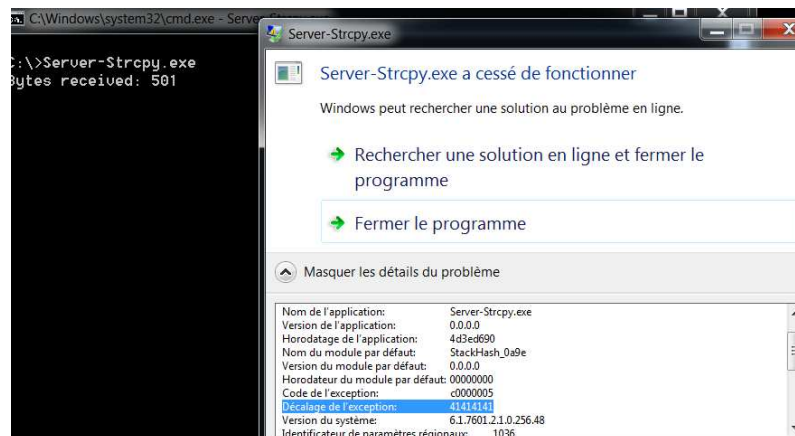
Ce n'est PAS un guide d'exploitation
Représente un cours en lui-même
Exploiter python avec Immunity Debugger
Analyse de binaire
Scripting et automatisation d'exploitations

Une formation
Alphorm.com



Une formation
Alphorm.com

Lab : Introduction



Merci



Une formation
Alphorm.com

Immunity Debugger



Hamza KONDAH



Une formation
Alphorm.com

Plan

Introduction

Options de Scripting

PyCommands

Lab : Immunity Debugger



Une formation
Alphorm.com

Introduction

Debugger
Outil légendaire
Développement d'exploit
Simple et puissant
Python ☺



Une formation
Alphorm.com

Options de Scripting

Pyhooks
PyCommands
PyScripts



PyCommands

APIs immlib

Exploitation au niveau du « runtime »

POC très facile

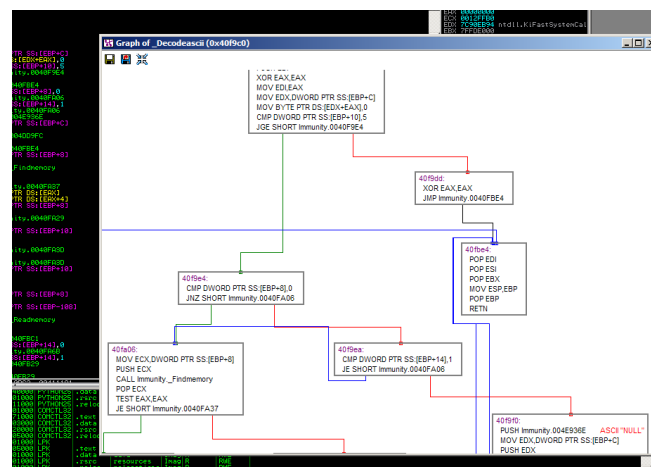
CLI ou GUI

Inclus dans IMD

Une formation
Alphorm
com



Lab : Immunity Debugger



Une formation
Alphorm
com

Merci



L'assemblage et le désassemblage



Hamza KONDAH



Une formation
Alphorm.com

Plan

Introduction

Lab : Assemblage et Désassemblage



Une formation
Alphorm.com

Introduction

Assemblage

`imm.assemble()`

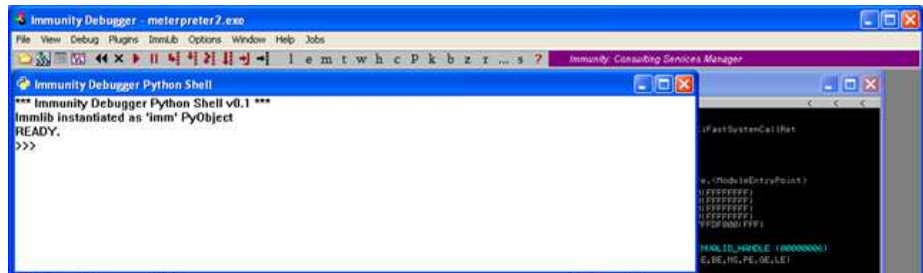
Instruction séparé
par « \n »

Désassemblage

`imm.disasm(address).getdisasm()`



Lab : Assemblage et Disas.



Une formation
Alphorm.com

Merci



Une formation
Alphorm.com

Pydasm



Hamza KONDAH



Une formation
Alphorm.com

Plan

Introduction

Lab : Pydasm



Introduction

Module Python → libdasm
Désassemblage de binaire
Différentes syntaxes (Intel, AT&T...)
Très intéressant à exploiter
Reverse Engineering

Une formation
Alphorm.com



Lab : PyDasm

```
[*] Routine #1
[*] Snapshot hook point - 0x100016c1
[*] Restore hook point - 0x10001593
[*] Argument - ESP+0

0x10005674 lea edi,[ecx-0xc3]
0x10005680 jmp 0x10005685
0x10005682 lea edi,[ecx-0xc4]
0x10005683 mov ecx,[esp+0xc]
0x10005689 test ecx,0xc1
0x1000568f jz 0x100056ae
0x10005691 mov dl,[ecx]
0x10005693 add ecx,0x1
0x10005696 test dl,dl
0x10005698 jz 0x10005700
0x1000569a mov [edi],dl
0x1000569c add edi,0xc1
0x1000569f test ecx,0xc1
0x100056a3 jmp 0x10005691
0x100056a7 jmp 0x100056aa
0x100056a9 mov [edi],edx <--- Crash
0x100056ab add edi,0x1
0x100056ae mov ebx,0x7efefeff
0x100056b3 mov eax,[ecx]
0x100056b5 add ebx,eax
0x100056b7 mov eax,0xffffffff
0x100056ba mov eax,edx
0x100056bc mov ebx,[ecx]
0x100056be add ecx,0x1
0x100056c1 test eax,0xb1010100
0x100056c6 jmp 0x100056a9
0x100056c8 test dl,dl
0x100056ca jz 0x10005700
0x100056cc test dh,dh
0x100056ce jz 0x100056ef
0x100056d0 test ebx,0x200000

Registers:
EAX=0x7efefefe
ECX=0x0011fc00 -> 77777777
EDX=0x12323232
EBX=0x00156de4 -> 77707777
ESP=0x001569e4 -> 07M32
EBP=0x00156978 -> 77777777
ESI=0x1011eecc -> ---
EDI=0x0011fffe -> Auth
EIP=0x100056a9

SEN:
Next SEN Record SE Handler Offset
0x12323232 0x323232 n/a
0xffffffff 0xffffffff n/a

Input Dump (28544 bytes):
[x36]x0d[x31]x0a[x65]x5d[x04]x29[x02]
[x6c]x05[x5d]x7e[x1a]x1e[x10]x35[x30]
[x0a]x2a[x54]x0b[x41]x2a[x65]x4b[x24]
[x2c]x79[x3d]x14[x42]x21[x5e]x27
[x6e]x22[x0a]x44[x7c]x10[x58]x23[x26]
[x21]x24[x05]x02[x1c]x38[x50]x05
[x0d]x7c[x34]x57[x3f]x6b[x1c]x37[x23]
[x71]x44[x59]x39[x4a]x47[x74]x33[x29]
[x1b]x6d[x74]x06[x37]x02[x1f]x1c[x50]
```

Une formation
Alphorm.com

Merci



PyDBG



Hamza KONDAH



Une formation
Alphorm.com

Plan

Introduction

Informations

Lab : PyDBG



Une formation
Alphorm.com

Introduction

Debugger en Python

Analyse du programme

Réduire le nombre de bug ^^

Inspection des process en cours



Informations

Variables

Dump de la
mémoire

Stack

Registres



Lab : PyDBG

```
1 from pydbg import *
2 from pydbg.defines import *
3 import os
4 import struct
5
6 os.system('CLS')
7 dbg = pydbg()
8
9 #Define target process
10 target_process = "acord32.exe"
11 pid_is_there = False
12
13 print "[*] Observing CreateFile(A-W)"
14
15
16 def handler_CreateFileW(dbg):
17     filename = ""
18     buffer_fileA = dbg.read_process_memory(dbg.context.Eip + 0x4, 4)
19     addr_filePointer = struct.unpack("L", addr_buffer_data[0])
20     filename = dbg.smart_dereference(addr_filePointer, True)
21     print "CreateFileW -> %s" % filename
22     return DBG_CONTINUE
23
24 def handler_CreateFileA(dbg):
25     offset = 0
26     buffer_fileA = ""
27     addr_buffer_data = dbg.read_process_memory(dbg.context.Eip + 0x4, 4)
28     addr_buffer_data = struct.unpack("L", addr_buffer_data[0])
29     buffer_fileA = dbg.smart_dereference(addr_buffer_data, True)
30     print "CreateFileA -> %s" % buffer_fileA
31     return DBG_CONTINUE
32
33 for pid,name in dbg.enumerate_processes():
34     if name.lower() == target_process:
35         pid_is_there = True
36         print "[*] Attaching to the process %s" % target_process
37         dbg.attach_pid(pid)
38         function2 = "CreateFileW"
39         function3 = "CreateFileA"
40         CreateFileW = dbg.func_resolve_debugger("kernel32.dll", "CreateFileW")
41         CreateFileA = dbg.func_resolve_debugger("kernel32.dll", "CreateFileA")
```

Merci



PyHooks



Hamza KONDAH



Une formation
Alphorm.com

Plan

Introduction

Lab :PyHooks



Une formation
Alphorm.com

Introduction

Dirigé par les events

Event Driven

Vue intrusive sur le fonctionnement
du programme

Approche réactive → Pendant l'
exécution

PyCommand ou stand alone



Introduction

FastLogHook
STDCALLFastLogHook
Hook
BpHook
LogBpHook
PreBpHook
AllExceptHook
PostAnalysisHook
AccessViolationHook
RunUntilAV
LoadDLLHook
UnloadDLLHook
CreateThreadHook
ExitThreadHook
CreateProcessHook
ExitProcessHook



Lab : PyHooks

```
import immlib
from immlib import AllExceptHook

class DemoHook (AllExceptHook) :

    def __init__(self) :
        AllExceptHook.__init__(self)

    def run(self, regs) :

        I
```

Merci



Le Fuzzing



Hamza KONDAH



Une formation
Alphorm.com

Plan

Introduction

Lab : Le fuzzing



Une formation
Alphorm.com

Introduction

Fuzzing ou *Fuzzy Test*

Envoi aléatoire de données

Services ou données

Crash du système

Analyse post crash



Lab : Le fuzzing

```
root@Pentesteur:~/Documents/fuzzing# ./bof_fuzz.py 192.168.133.133 21 A 100 1000
Enter ftp username: anonymous
Enter ftp email: mail@mail.mail
Enter FTP command to fuzz: MKD
Sending 100 instances of payload (A) to target
Sending 200 instances of payload (A) to target
Sending 300 instances of payload (A) to target
Sending 400 instances of payload (A) to target
Sending 500 instances of payload (A) to target
Sending 600 instances of payload (A) to target
Sending 700 instances of payload (A) to target
Sending 800 instances of payload (A) to target
Sending 900 instances of payload (A) to target
Sending 1000 instances of payload (A) to target

There is no indication that the server has crashed
```

Une formation
Alphorm.com

Merci



Une formation
Alphorm.com

Exploitation de vulnérabilités



Hamza KONDAH



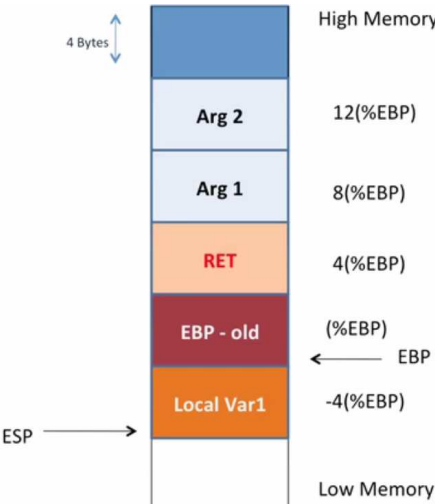
Une formation
Alphorm.com

Plan

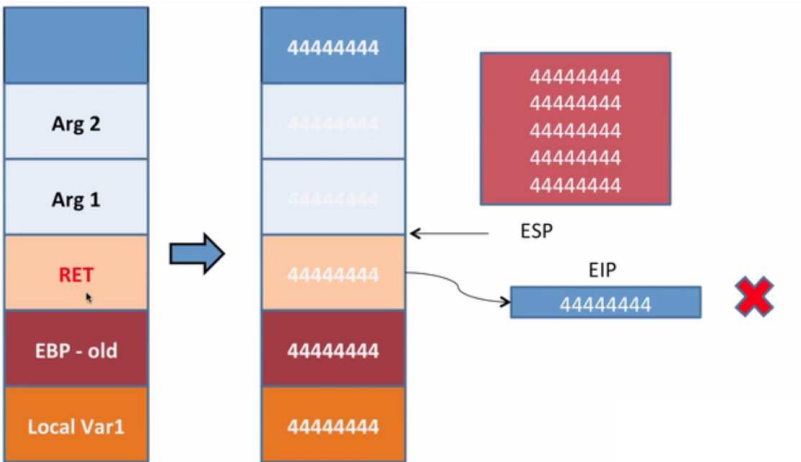
Segmentation mémoire
Exploitation de Buffer Overflow
Lab : Exploitation de BoF



Segmentation mémoire

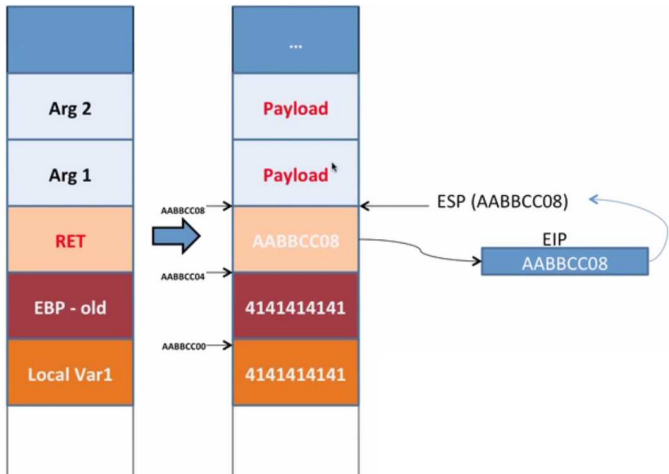


Exploitation de BoF

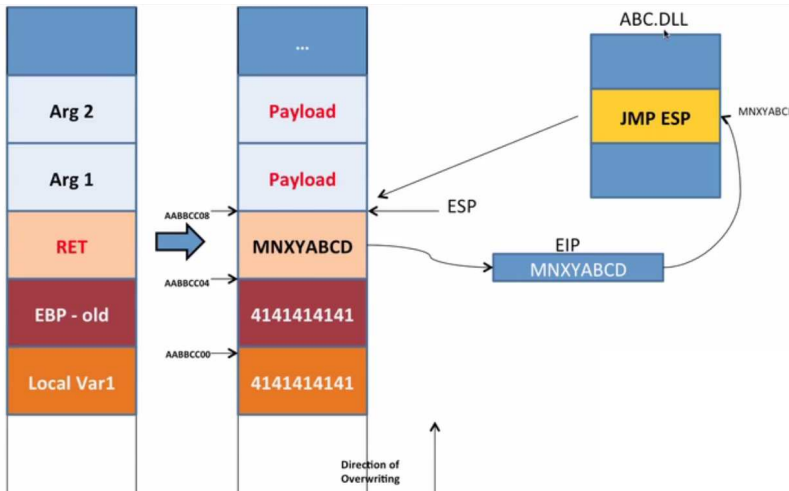




Exploitation de BoF



Exploitation de BoF



Merci



Introduction au forensique



Hamza KONDAH



Une formation
Alphorm.com

Plan

Définition

Méthodologie



Une formation
Alphorm.com

Définition

Analyse post-mortem

Procédures légales

Recherche de preuves

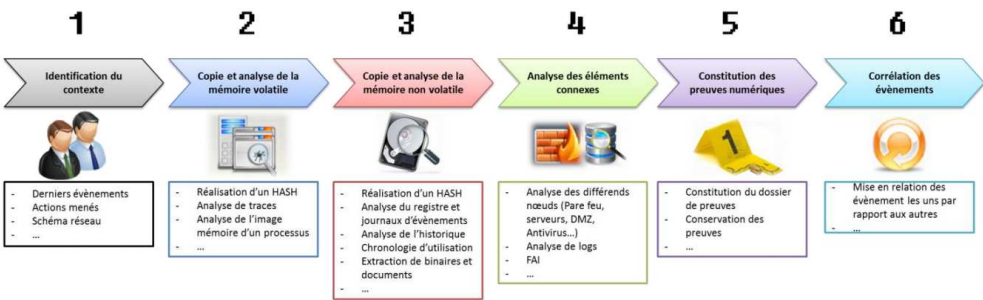
Preuves « Acceptables »

Techniques de Hacking

Sécurité offensive



Méthodologie



Merci



Une formation
Alphorm.com

La Cryptographie



Hamza KONDAH



Une formation
Alphorm.com

Plan

Introduction

Le Hashage

Base 64

ROT13

Lab : Python et Cryptographie



Introduction

Rendre l'information illisible par un tiers

Limite des outils traditionnels

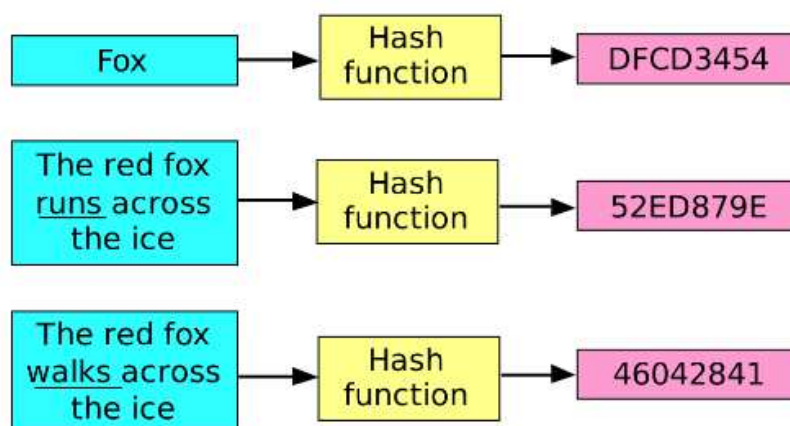
besoin de customisation

Environnement de Pentest ☺

Une formation
Alphorm.com



Le Hashage



Une formation
Alphorm.com



Le Hashage

14.1. hashlib — Secure hashes and message digests

New in version 2.5.

Source code: Lib/hashlib.py

This module implements a common interface to many different secure hash and message digest algorithms. Included are the FIPS secure hash algorithms SHA1, SHA224, SHA256, SHA384, and SHA512 (defined as well as RSA's MD5 algorithm (defined in internet RFC 1321). The terms secure hash and message digest are interchangeable. Older algorithms were called message digests. The modern term is secure hash.

Note: If you want the `adler32` or `crc32` hash functions, they are available in the `zlib` module.

Warning: Some algorithms have known hash collision weaknesses, refer to the "See also" section at the end.

There is one constructor method named for each type of *hash*. All return a hash object with the same simple interface. For example, use `sha1()` to create a SHA1 hash object. You can now feed this object with `update()` method. At any point you can ask it for the *digest* of the concatenation of the strings fed to it so far using the `digest()` or `hexdigest()` methods.

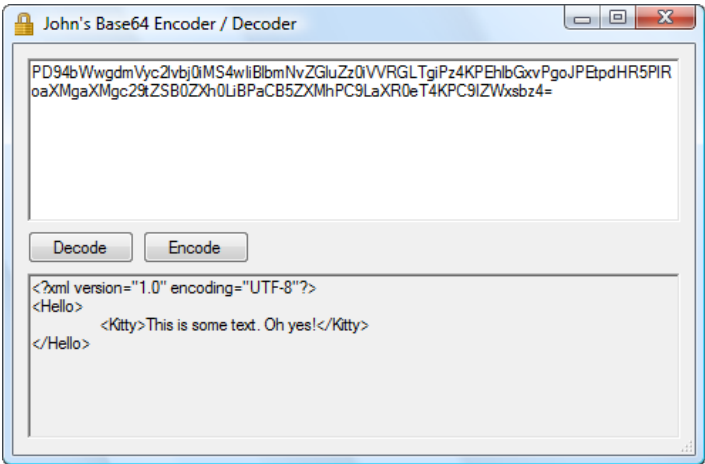
Constructors for hash algorithms that are always present in this module are `md5()`, `sha1()`, `sha224()`, `sha256()`, `sha384()`, and `sha512()`. Additional algorithms may also be available depending upon the OpenSSL library on your platform.

For example, to obtain the digest of the string "Nobody inspects the spammish repetition":

```
>>> import hashlib
>>> m = hashlib.md5()
>>> m.update("Nobody inspects")
>>> m.update(" the spammish repetition")
>>> m.digest()
'\x00\x0c\x08\xdd\x1e\x05\x09\x0d\x09\x01\x0d\x0f\xff\x09'
>>> m.digest_size
16
>>> m.block_size
64
```



Base 64





ROT13

Rotate by 13 places

Chiffre de César

Chiffrement de texte

Décalage de 13 texte

Une formation
Alphorm.com



Lab: La Cryptographie

```
from cryptography.hazmat.backends import default_backend
from cryptography.hazmat.primitives import hashes
from cryptography.hazmat.primitives.asymmetric import padding

message = b"The SECRET KEY"
ciphertext = public_key.encrypt(
    message,
    padding.OAEP(
        mgf=padding.MGF1(algorithm=hashes.SHA1()),
        algorithm=hashes.SHA1(),
        label=None))

# The same as encryption, but using the private key instead
plaintext = private_key.decrypt(
    ciphertext, ...)
```

Une formation
Alphorm.com

Merci



Les Métadonnées



Hamza KONDAH



Une formation
Alphorm.com

Plan

Introduction

Lab : Les Métadonnées



Une formation
Alphorm.com

Introduction

Données essentielles pour le
forensique

Path, date de création, utilisateurs ...

Extraction rapide

Librairie dédiée

MP3, PDF, Image ...



Une formation
Alphorm.com

Lab : Les Métadonnées

```
oioio.py x
1  import Image, os, sys
2
3  image_file = raw_input("[*] Enter Image Path: ")
4
5  if os.path.isfile(image_file):
6
7      directory, filename = os.path.split(image_file)
8
9      image = Image.open(image_file)
10
11      data = list(image.getdata())
12
13      image_without_exif = Image.new(image.mode, image.size)
14
15      image_without_exif.putdata(data)
16
17      image_without_exif.save(directory + "/Exif_Stripped_" + filename)
18
19      print("[*] File Saved: %s/Exif_Stripped_%s" % (directory, filename))
20      sys.exit(0)
21  else:
22      print("[*] Image Path Does Not Exists !!!")
23      sys.exit(1)
```

Merci



Une formation
Alphorm.com

Les fichiers ZIP



Hamza KONDAH



Une formation
Alphorm.com

Plan

Introduction

Lab : Fichiers ZIP



Une formation
Alphorm.com

Introduction

Souvent protégé par un mot de passe

Les outils ne manquent pas
d'efficacité

Besoin de customisation

Bruteforcing ou attaque par
dictionnaire



Une formation
Alphorm.com

Lab : Fichiers ZIP

```
1 import zipfile
2
3
4 def main():
5     """
6     Zipfile password cracker using a brute-force dictionary attack
7     """
8     zipfilename = 'test.zip'
9     dictionary = 'dictionary.txt'
10
11     password = None
12     zip_file = zipfile.ZipFile(zipfilename)
13     with open(dictionary, 'r') as f:
14         for line in f.readlines():
15             password = line.strip('\n')
16             try:
17                 zip_file.extractall(pwd=password)
18                 password = 'Password found: %s' % password
19             except:
20                 pass
21     print password
22
23 ~
```

Merci



Les fichiers office



Hamza KONDAH



Une formation
Alphorm.com

Plan

Introduction

Lab : Les fichiers office



Une formation
Alphorm.com

Introduction

Les documents office contiennent
souvent des informations importantes
Principalement : Des macros ou
exploits

Manque d'outils

Analyse forensique

Fichiers endommagés



Lab : Les fichiers office

```
#!/usr/bin/env python
import os, fnmatch
def tous_les_fichiers( racine, motifs='*', un_seul_niveau=False,
    repertoires = False):
    motifs=motifs.split(';')
    for chemin, sous_reps, fichiers in os.walk(racine):
        if repertoires:
            fichiers.extend(sous_reps)
        fichiers.sort()
        for nom in fichiers:
            for motif in motifs:
                if fnmatch.fnmatch(nom,motif):
                    yield os.path.join(chemin,nom)
                    break
        if un_seul_niveau:
            break
    for chemin in tous_les_fichiers('/tmp','*.py ;*.htm ;*.html') :
        print chemin
```

Une formation
Alphorm.com

Merci



Une formation
Alphorm.com

Les Mails



Hamza KONDAH



Une formation
Alphorm.com

Plan

Introduction

Lab : Les Mails



Une formation
Alphorm.com

Introduction

Recherche de mail

Recherche dans une boîte mail

Méthodologie à customiser

A voir : Formation forensique



Une formation
Alphorm.com

Lab : Les Mails

```
try:
    mail = imaplib.IMAP4_SSL(SMTP_SERVER)
    mail.login(FROM_EMAIL, FROM_PWD)
    mail.select('inbox')

    type, data = mail.search(None, 'ALL')
    mail_ids = data[0]

    id_list = mail_ids.split()
    first_email_id = int(id_list[0])
    latest_email_id = int(id_list[-1])

    for i in range(latest_email_id, first_email_id, -1):
        typ, data = mail.fetch(i, '(RFC822)' )
```

Merci



La Stéganographie



Hamza KONDAH



Plan

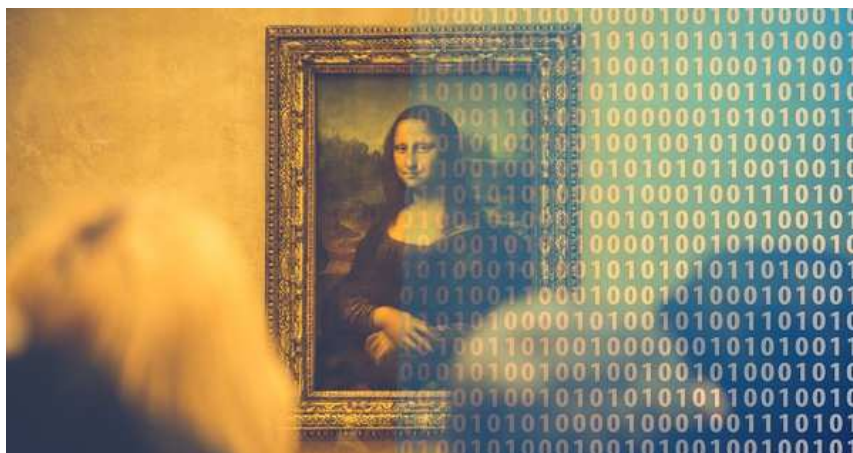
Introduction

Lab : La Stéganographie

Une formation
Alphorm.com



Introduction



Une formation
Alphorm.com



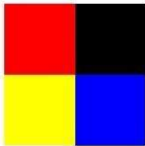
Introduction

Color Chart	R	G	B	Color Name
	0	0	0	Black
	255	255	255	White
	224	224	224	Light Gray
	128	128	128	Gray
	64	64	64	Dark Gray
	255	0	0	Red
	255	96	208	Pink
	160	32	255	Purple
	80	208	255	Light Blue
	0	32	255	Blue
	96	255	128	Yellow-Green
	0	192	0	Green
	255	224	32	Yellow
	255	160	16	Orange
	160	128	96	Brown
	255	208	160	Pale Pink



Introduction

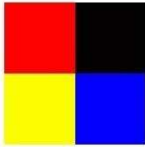
Original Image



11111111	00000000
00000000	00000000
00000000	00000000
11111111	00000000
11111111	00000000
00000000	11111111

Least Significant Bit
Steganography

Stego Image

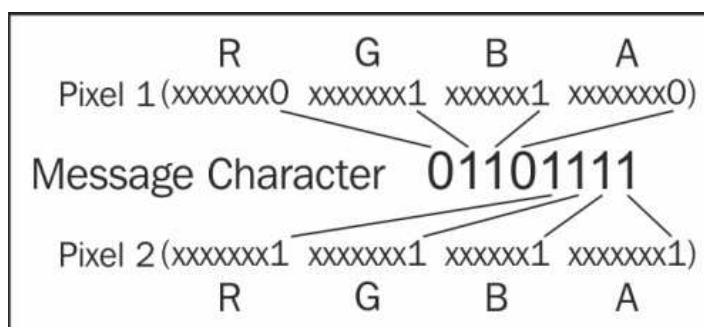


11111101	00000011
00000010	00000001
00000000	00000010
11111100	00000011
11111101	00000001
00000001	11111100

} **c** **a** **t**
01 10 00 11 01 10 00 01 01 11 01 00



Introduction



Une formation
Alphorm.com



Lab : Stéganographie

```
img = img.convert('RGBA')
datas = img.getdata()

newData = []
digit = 0
temp = ''
for item in datas:
    if (digit < len(binary)):
        newpix = encode(rgb2hex(item[0], item[1], item[2]), binary[digit])
        if newpix == None:
            newData.append(item)
        else:
            r, g, b = hex2rgb(newpix)
            newData.append((r, g, b, 255))
            digit += 1
    else:
        newData.append(item)
img.putdata(newData)
img.save(filename, "PNG")
return "Completed!"
return "Incorrect Image mode, couldn't hide"

def retr(filename):
```

Une formation
Alphorm.com

Merci



Volatility



Hamza KONDAH



Une formation
Alphorm.com

Plan

Introduction

Fonctionnalités

Lab : Volatility



Une formation
Alphorm.com

Introduction

Analyse de la RAM

Forensics

Connexions réseau, mots de passe,
fichiers effacés

Plus grand projet Opensource pour le
forensics



Une formation
Alphorm.com

Fonctionnalités

Affichage de la liste des connexions réseau
ouvertes et scan des objets de connexion

Affichage de la liste des DLL chargées par
chaque processus

Affichage des fichiers ouverts pour chaque
processus

Réalisation de différents dump



Une formation
Alphorm.com

Fonctionnalités

Identification de la propriété des
images incluant les données, l'horaire,
la date et le lieu

Affichage d'une liste des clés de
registre pour chaque processus trouvé
dans la table des processus



Une formation
Alphorm.com

Lab : Volatility

```
=====
Volatility Framework - Volatile memory extraction utility framework
=====
```

The Volatility Framework is a completely open collection of tools, implemented in Python under the GNU General Public License, for the extraction of digital artifacts from volatile memory (RAM) samples. The extraction techniques are performed completely independent of the system being investigated but offer visibility into the runtime state of the system. The framework is intended to introduce people to the techniques and complexities associated with extracting digital artifacts from volatile memory samples and provide a platform for further work into this exciting area of research.

The Volatility distribution is available from:
http://www.volatilityfoundation.org/#!/releases/component_71401

Volatility should run on any platform that supports
Python (<http://www.python.org>)

Merci



Une formation
Alphorm.com

Conclusions et Perspectives



Hamza KONDAH



Une formation
Alphorm.com

Bilan

Notions de base
Sécurité des réseaux
Le web et python
Le développement d'exploits
Le forensique
La cryptographie



A Faire

Faire les exercices au niveau des
ressources

Solutions disponibles

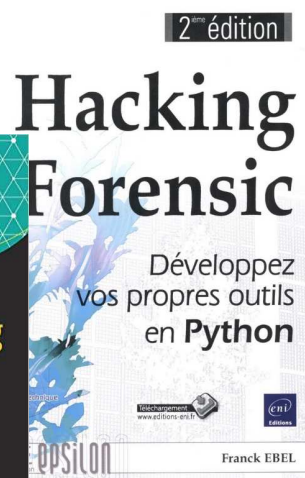
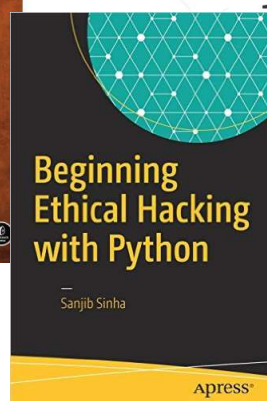
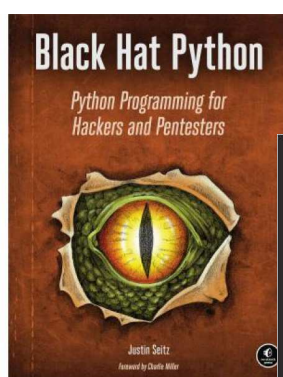
Lire ☺

Pratiquez !

Une formation
Alphorm.com



Bibliographie



Une formation
Alphorm.com



Formations à suivre



Hacking & Sécurité, Expert : Les vulnérabilités des Réseaux

Hamza KONDAH

Note : ★★★★★ Vues : 198.394

17%



Hacking et Sécurité, Expert : Les vulnérabilités Web

Hamza KONDAH

Note : ★★★★★ Vues : 130.336

2%



Hacking et Sécurité, Expert : Metasploit

Hamza KONDAH

Note : ★★★★★ Vues : 107.609

20%



Hacking et Sécurité, Expert : Réseaux sans Fil

Hamza KONDAH

Note : ★★★★★ Vues : 207.494

54%



Hacking et Sécurité : Maîtriser les techniques avancées

Hamza KONDAH

Note : ★★★★★ Vues : 474.765



Hacking et Sécurité : Acquérir les fondamentaux

Hamza KONDAH

Note : ★★★★★ Vues : 1.194.352

Une formation
Alphorm.com



Prochainement











Une formation
Alphorm.com



Merci