Securitate Software

XII Penetration testing

Scop și Motivație

- Sistemele informatice sunt inerent vulnerabile
- Vulnerability: „System vulnerability is defined to be the intersection of a system susceptibility or flaw, access to the flaw, and the capability to exploit the flaw" [1]
  - Prezența vulnerabilității
  - Identificarea vulnerabilității
  - Exploatarea vulnerabilității
- Exploatarea vulnerabilităților de către criminalii informatici poate cauza o mulțime de inconveniențe
  - Pierderi financiare, de informații confidențiale, de vieți omenești
- Este necesară identificarea și repararea vulnerabilităților înainte ca acestea să fie exploatate de atacatori: pentesting

Scop și Motivație (II)

- Pentesting vs. Hacking
  - Accesul, **fără drept**, la un sistem informatic [...]
    - Permisiunea / autorizația
- Pentesting vs. Vulnerability research
  - Pentesting - în general se rezumă la vulnerabilități cunoscute
  - Vulnerability research – căutare de noi vulnerabilități (Project Zero)
- De ce pentesting?
  - Bug bounties
  - Carieră – it pays well
  - E legal & fun

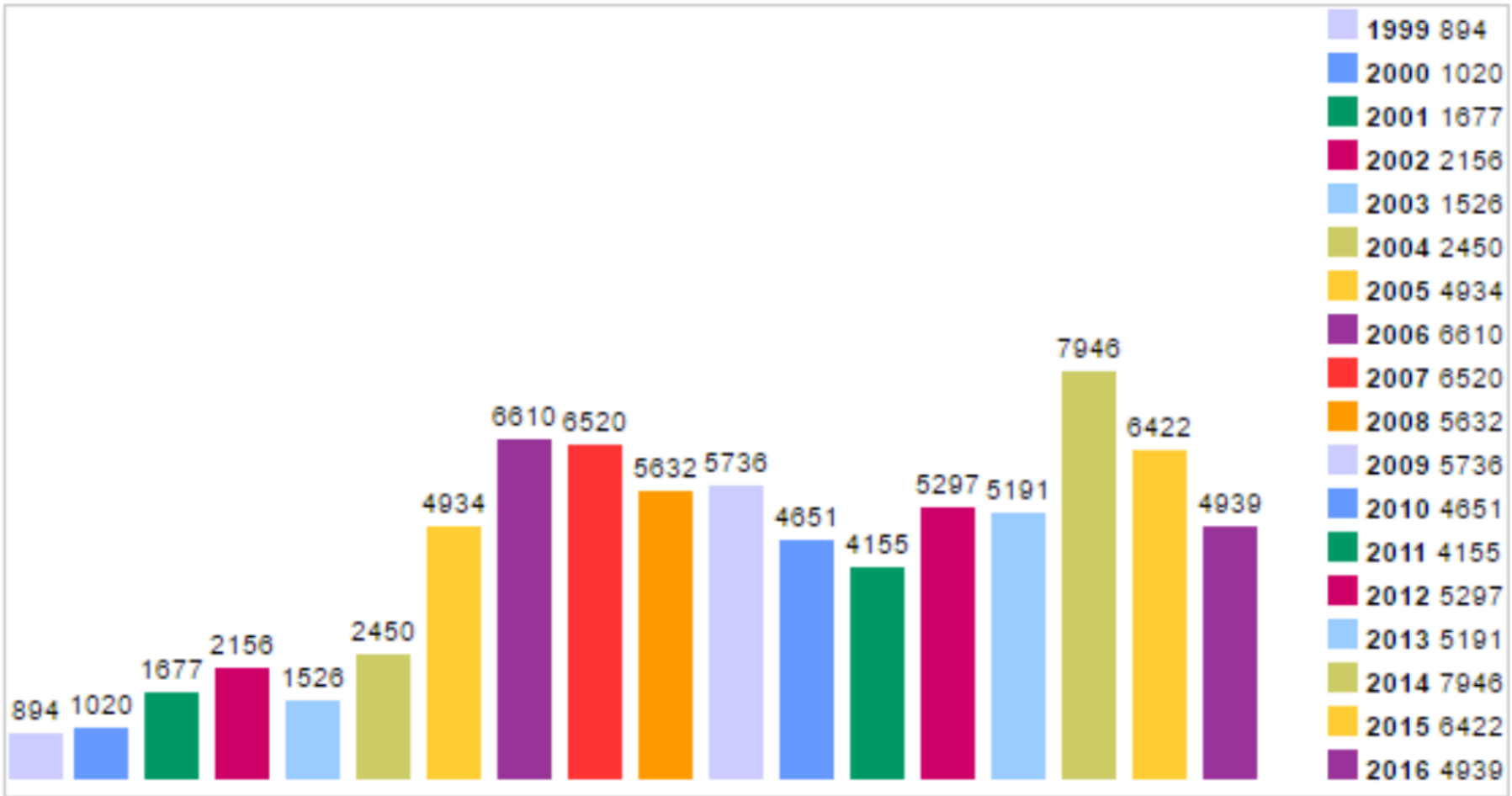
Scop și Motivație (III)

- Securitatea e deseori ignorată
  - Usability = 1 / Security
  - Mult cod, mulți programatori
    - Calitate îndoielnică...
- Soluțiile teoretice sunt insuficiente
  - Crypto, parole, etc
    - Tell this to Flash :)
- Atacurile contemporane sunt focalizate pe câștig financiar sau spionaj
  - Unele organizații chiar devin preocupate de securitate!

Scop și Motivație (IV)

- Vulnerabilități noi/necunoscute
  - CVE – Common Vulnerabilities and Exposures
  - CVD – Coordinatied Vulnerability Disclosure
  - Notifcați vendorul!
    - scăpați de eventuale probleme legale
    - vi se recunoaște meritul descoperirii vulnerabilității
  - Puteți trăi din asta (`https://bugcrowd.com/list-of-bug-bounty-programs`)
  - Fiți responsabili!
- De cele mai multe ori, vă veți baza pe vulnerabilități cunoscute
  - `https://www.exploit-db.com`

Scop și Motivație (V) - număr vulnerabilități pe an

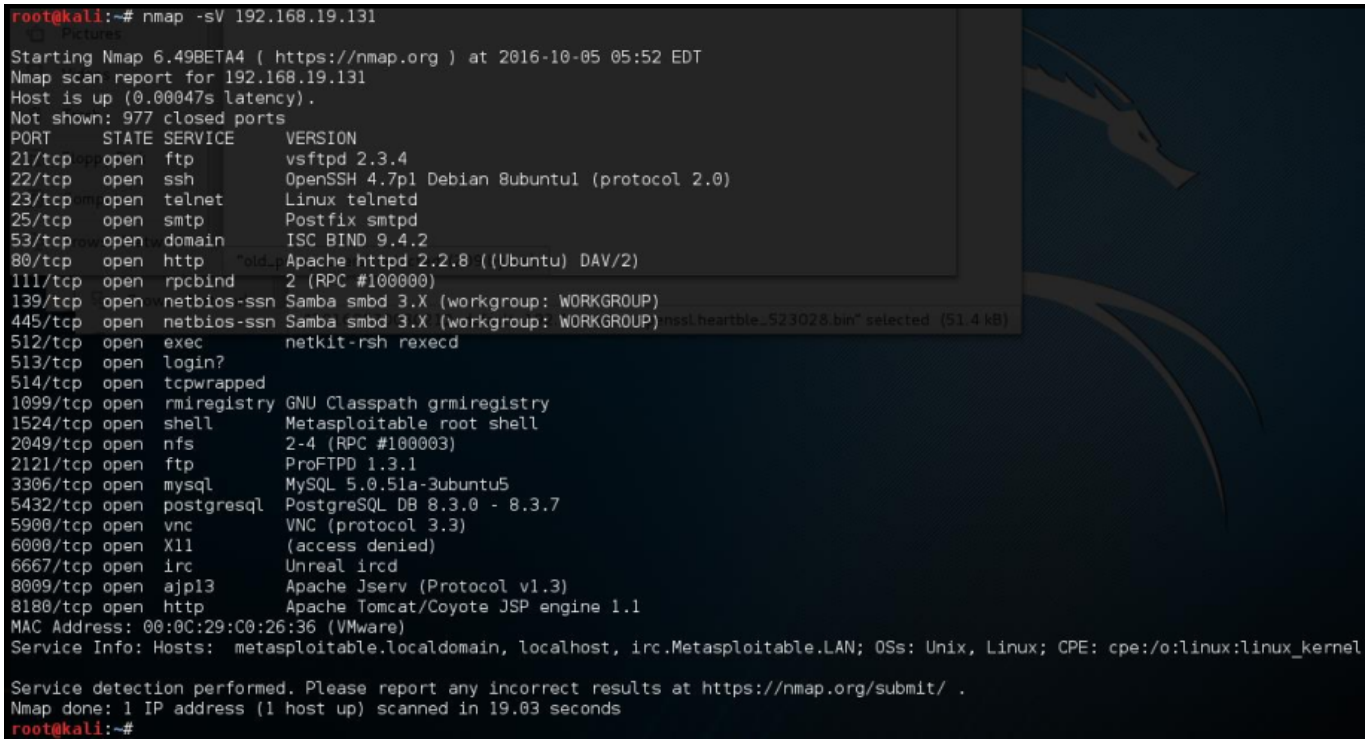


Metodologie

- Obținerea de informații
- Cel mai important pas
- Multe informații publice

Metodologie (II)

- Enumerare servicii
  - Port scanning, SNMP, DNS, SMTP, SQL etc



Scop

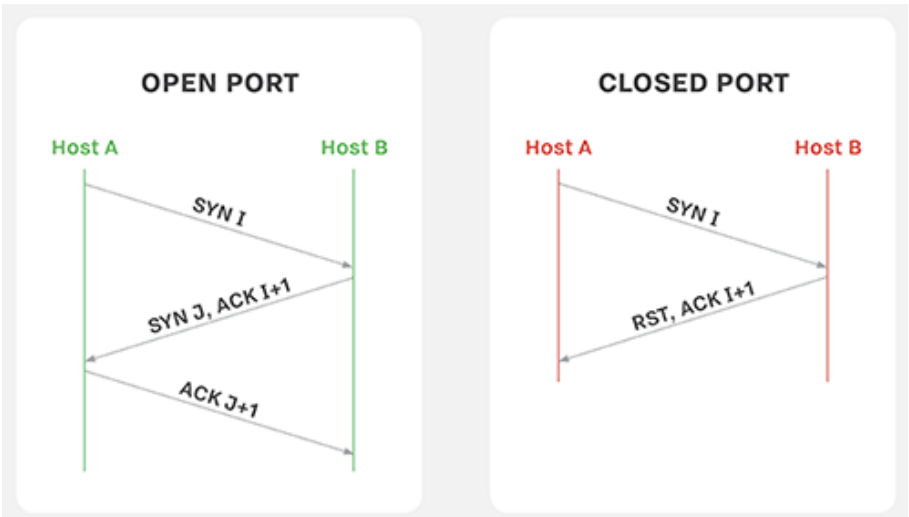
- Identificarea stării porturilor scanate (port scan)
  - Deschis – există un serviciu activ care acceptă conexiuni
  - Închis – nu există nici un serviciu activ care acceptă conexiuni
  - Filtrat – portul este filtrat de către un firewall
- Identificarea serviciilor active pe sisteme (service scan)
  - ftp, http, smtp, ...

Scop (II)

- Cel mai important pas după culegerea de informații
  - În această fază se descoperă servicii active și/sau vulnerabile
- Oferă informații importante cu privire la rolul fiecărui sistem din rețea
- Implică identificarea atât a porturilor deschise, cât și a serviciilor care rulează pe ele
- Port scan vs. port sweep
  - Port scan = scanează o mulțime de porturi pe un sistem dat
  - Port sweep = scanează pe o mulțime de sisteme un port dat

Tehnici de scanare a porturilor - SYN scan

- 1. SYN scan
  - Se trimite un pachet cu SYN setat
  - Rapidă, poate scana mii de porturi/secundă
  - Relativ stealth
  - „Half open scanning” – nu se deschide o conexiune completă, ci se trimite doar un SYN:



  - port filtrat: nu se primește răspuns (ICMP unreachable)

Tehnici de scanare a porturilor - Connect scan

- 2. Connect scan
  - Spre deosebire de SYN scan, se deschide o conexiune completă
  - Nu are nevoie de privilegii speciale pentru a trimite pachete raw
  - Se bazează pe sistemul de operare pentru stabilirea conexiunii
  - Durează mult mai mult
  - Ușor de prins de IDS-uri
  - De preferat SYN scan când e posibil

Tehnici de scanare a porturilor - NULL scan

- 3. NULL scan, FIN scan, Xmas scan
  - NULL scan: se trimite un pachet cu toate flag-urile 0
  - FIN scan: se trimite un pachet cu FIN setat
  - Xmas scan: se trimite un pachet cu FIN, PSH, URG setați
  - Uneori, mai stealth decât SYN scan
  - Funcțional, sunt identice
  - Răspuns la NULL, FIN și Xmas scan:
    1. port deschis: fără răspuns
    2. port închis: RST
    3. port filtrat: fără răspuns (ICMP unreachable)

Tehnici de scanare a porturilor - ACK scan

- 4. ACK scan
  - Se trimite un pachet cu ACK setat
  - Determină comportamentele firewall-urilor
  - Nu determină dacă un port este deschis/închis, ci dacă e filtrat/nefiltrat
  - Răspuns la ACK scan:
    1. port nefiltrat: RST
    2. port filtrat: fără răspuns (ICMP unreachable)

Tehnici de scanare a porturilor - Window scan

- 5. Window scan
  - Utilizează câmpul Window pentru a diferenția porturile deschise de cele închise (în rest identică cu ACK scan)
  - Se bazează pe anumite implementări de TCP
  - nu este de încredere
  - Răspuns la Window scan:
    1. port deschis: Window >0
    2. port închis: Window $== 0$

Tehnici de scanare a porturilor - maimon scan

- 6. maimon scan
  - Identică cu NULL, FIN, Xmas, dar folosește FIN/ACK
  - Răspuns la Maimon scan:
    1. port deschis: fără răspuns
    2. port închis: RST

Tehnici de scanare a porturilor - UDP scan

- 7. UDP scan
  - Multe servicii folosesc protocolul UDP (DHCP, DNS, SNMP, ...)
  - Lentă
  - Trebuie, totuși, avută în vedere în procedeul de pentest!

Tehnici de identificare a serviciilor

- Un port deschis este inutil, dacă nu știm ce serviciu rulează pe el
- Pasul de identificare a serviciilor și a versiunilor este critic!
- Informații ce se pot extrage:
  - Protocolul (FTP, SMTP, HTTP, etc.)
  - Aplicația (Apache, ProFTPd, Postifx, etc.)
  - Versiunea
  - Numele mașinii
  - Tipul device-ului
  - Sistemul de operare

Tehnici de identificare a serviciilor

- rezultate nmap

```
# nmap -sV -p21,22,80 10.10.10.XX
Nmap scan report for 10.10.10.XX
Host is up, received user-set (0.045s latency).
Scanned at 2017-12-08 16:07:54 EET for 827s

PORT   STATE SERVICE  VERSION
21/tcp open  ftp      vsftpd 3.0.3
22/tcp open  ssh      OpenSSH 7.5 (protocol 2.0)
80/tcp open  http     Apache httpd 2.4.27 ((Unix))
```

Tehnici de identificare a serviciilor

- Nmap identifică în mod automat serviciile care rulează pe un anumit port
  - Poate identifica și alte informații, precum sistemul de operare
- Uneoi, se poate utiliza „identificarea manuală”, citind bannerul

```
# nc -nv 10.10.10.XX 2222
(UNKNOWN) [10.10.10.XX] 2222 (?) open
SSH-2.0-OpenSSH_7.2p2 Ubuntu-4ubuntu2.2
```

- Uneori, identificarea manuală sau nmap nu ajută
  - Protocoale/servicii custom
  - Nu se poate face mare lucru
  - Exceptând reverse-engineering

Tehnici de identificare a serviciilor

- Identificarea sistemului de operare
- Nmap e capabil să identifice în mod automat sistemul de operare
  - Trimite pachete TCP/UDP și inspectează fiecare bit din răspuns
  - Fiecare OS are particularitățile sale în stiva TCP/IP
- Uneori identificarea e directă (din servicii)

```
# curl -i http://10.10.10.XX
HTTP/1.1 200 OK
Date: Sun, 05 Nov 2017 10:08:52 GMT
Server: Apache/2.4.18 (Ubuntu)
Last-Modified: Fri, 22 Sep 2017 20:01:19 GMT
ETag: "89-559ccac257884"
Accept-Ranges: bytes
Content-Length: 137
Vary: Accept-Encoding
Content-Type: text/html

<!DOCTYPE html>
<html>
<body>
```

Evitarea Detectiei

- Scanarea de porturi e relativ ușor detectabilă de firewall-uri
- Sursele de port-scans pot să fie deconectate automat de la rețea, banate, etc.
- Este important ca scanarea să fie stealth

Evitarea Detectiei (II)

1. Fragmentarea pachetelor
   - Pachetele TCP sunt fragmentate
   - Multe firewall-uri nu tratează acest caz (performanță...)
   - Unele tratează acest caz, făcând tehnica inutilă
2. Decoys
   - Pentru fiecare port scanat, se trimit n request-uri, fiecare request având o adresă IP sursă spoofed
   - Victima vede că n+1 sisteme o scanează, dar nu știe care e sursa reală a atacului
   - Se poate mitiga făcând router path tracing sau prin blocarea spoofingului

Evitarea Detectiei (III)

3. Spoofing
   - Se utilizează altă sursă pentru scanare; exemplu: FTP Bounce Back
4. Proxy
   - Scanarea se face prin unul sau mai multe proxy-uri
   - Victima vede că este scanată de ultimul proxy dintr-un asemenea lanț
5. Alte tehnici
   - Checksum invalid, MAC spoof, TTL modificat, apendare de date la pachetele trimise, etc.

Mitigari

- Nu există o soluție reală pentru a preveni port-scan...
- Majoritatea entităților legale consideră o scțiune de tip port scan ca fiind legală (exceptând cazul în care îl folosim pentru a exploata un serviciu)
- Opriți orice serviciu nu e necesar
  - Un server HTTP nu are, în mod ideal, nevoie de un server FTP
- Diferite tool-uri de „sercuritate”
  - PortSentry, TCP Wrappers, etc

Utilitarul NMAP

- `https://nmap.org`
- Unealtă indispensabilă oricărui pentester
- Orice pentester trebuie să ajungă să stăpânească acest tool
- Foarte complex
  - 114 linii de help
  - `https://nmap.org/book/man-briefoptions.html`
- Câteva din cele mai importante capabilități sunt descrise în cele ce urmează

Utilitarul NMAP (II)

- Target specification
  - Poate scana câte un host, sau rețele - atât după nume, cât și după adresă
  - Țintele de scanat pot fi oferite într-un fișier de intrare
- Host discovery
  - Suportă ping scan, port scan, port sweep
  - Suportă DNS-uri custom
- Scan techniques
  - Suportă toate tehnicile enumerate în curs + idle, SCTP, IP, FTP bounce scan
- Port specification
  - Default, scanează cele mai comune 1000 porturi
  - Se pot specifica intervale custom

Utilitarul NMAP (III)

- Service/Version detection
  - Detectează automat serviciile și versiunile care rulează pe fiecare port
- Script scan
  - Se pot crea scripturi dedicate de scanare
- OS detection
  - Detectează sistemul de operare
- Timing and performance
  - Suportă scanare paralelă (multithreaded)
  - Packets throtling

Utilitarul NMAP (IV)

- Firewall/IDS evasion and spoofing
  - Suportă toate tehnicile de evasing prezentate în curs (+)
- Output în multiple formate
  - XML, grepable

Introducere

- Pas 1: info gathering
- Pas 2: enumerare de servicii
- Pas 3: exploatare
- Pas 4: persistență
- Pas 5: enumerare post-atac
- Pas 6: cleanup

Metodologie (III)

- Exploatare/penetrare
  - Obținerea accesului la sistemul vizat
    - ... de obicei folosind exploit-uri pentru vulnerabilități descoperite la pasul precedent

```
msf exploit(vsftpd_234_backdoor) > exploit

[*] Banner: 220 (vsFTPd 2.3.4)
[*] USER: 331 Please specify the password.
[+] Backdoor service has been spawned, handling...
[+] UID: uid=0(root) gid=0(root)
[*] Found shell.
[*] Command shell session 2 opened (192.168.19.159:45371 -> 192.168.19.131:6200) at 2016-10-05 06:16:06 -0400

whoami
root
id
uid=0(root) gid=0(root)
```

Exploatarea

- „Exploit” – secvență de instrucțiuni sau date care profită de pe urma unei vulnerabilități, pentru a obține un comportament inadecvat
  - Exemplu: Remote Code Execution (RCE)
  - Exemplu: Local Privilege Escalation (LPE)
  - Exemplu: Information Disclosure (ID)
  - Exploit != payload
- Motivul pentru care facem pentesting
  - Prezența vulnerabilității este în general suficientă pentru a lua măsuri
  - Dacă vulnerabilitatea poate să fie și exploatată, cu atât mai bine
- Pasul care (probabil) oferă control asupra sistemului vulnerabil sau duce la compromiterea organizației

Exploatarea (II)

- Vulnerabilitățile trebuie să fie deja identificate
  - Info gathering și service enumeration
- În această fază, trebuie să știm exact ce exploit să folosim
  - Trebuie doar să alegem payload-ul potrivit, în funcție de exploit și în funcție de ce dorim să obținem, în funcție de sistemul de operare și de hardware
- În general, aplicăm principiul „lowest hanging fruit” – exploatăm vulnerabilitățile cele mai critice
  - Urmărim să obținem control asupra sistemului vulnerabil
    - Nu neapărat cu o vulnerabilitate de tip RCE!
  - O dată compromis un sistem, putem face lateral movement

Exploatarea (III)

- Pentru a exploata o vulnerabilitate, avem nevoie de un exploit
- Surse de exploit-uri:
  - Metasploit
  - exploit-db
  - packetstormsecurity
  - Dezvoltare proprie/exploatare manuală
- De cele mai multe ori, vom găsi exploit-uri în Metasploit sau pe exploit-db
  - Dar nu întotdeauna
- În funcție de preferințele clientului, se pot dezvolta exploit-uri custom
  - necesită aptitudini și timp...

Tipuri de vulnerabilități

- Tipuri de exploit-uri == tipuri de vulnerabilități
- Fiecare vulnerabilitate este exploatată într-un fel
- Chiar și vulnerabilități identice pot fi exploatate diferit, în funcție de aplicație, condiții specifice, etc.
  - Exemplu: use-after-free în Internet Explorer vs. use-after-free într-un image viewer
- Un exploit depinde foarte mult și de mitigările de pe sistemul țintă
  - Bypass de ASLR, bypass DEP/NX, bypass CFG, etc

Tipuri de vulnerabilități (II)

- Binare:
  - Coruperi de memorie, Race-conditions, Input nevalidat corespunzător, etc.
- Web:
  - XSS, CSRF, LFI, RFI, RCE, SQLi, etc.
- Generice:
  - Credențiale slabe/default, ID, arbitrary directory traversals, etc.
- Rezultat:
  - RCE – remote code execution
  - NON-RCE – nu oferă remote code execution

Tipuri de payload-uri

- În general, vorbim de „payload” în cazul RCE
- Payload-ul reprezintă codul care se va executa ca urmare a exploatării
  - În general, acesta este un „shellcode”
- Uneori, se utilizează shellcode-uri „staged” – mai multe stagii
  - Dacă shellcode-ul este foarte mare
- Uneori, se utilizează un atac de tip „drive-by-download” – shellcode-ul downloadează și execută un binar

Tipuri de payload-uri (II)

- Metasploit conține o listă uriașă de payload-uri
- Fiecare payload se pretează la un anumit tip de exploit, sistem de operare și arhitectură de procesor (în cazul celor binare)
- În funcție de OS, metasploit oferă payload-uri pentru:
  - Windows x86/x64
  - OSX
  - Solaris
  - Linux
- De asemenea, există payload-uri agnostice:
  - Python
  - PHP
  - Java
  - Perl

Tipuri de payload-uri (III)

Payload-uri binare

- Utilizate pentru a exploata vulnerabilități binare
- Windows, Linux, OSX, Android, etc.
- X86, x64, MIPS, ARM, SPARC, etc.
- Alegeți cu grijă payload-ul în funcție de OS și CPU!
  - Pe un router, probabil veți folosi un payload MIPS pentru Linux
  - Pe un smartphone, probabil veți folosi un payload ARM pentru Android
- Alegeți payload-ul în funcție de ce doriți să faceți

Tipuri de payload-uri (IV)

Payload-uri generice / scripturi / comenzi:

- Utile în special în sfera Web
- PHP, Ruby, Perl, Python, etc.
- Trebuie alese în funcție de interpretorul de scripturi vizat

Tipuri de payload-uri (V)

- Principalul criteriu de alegere al payload-ului este efectul dorit
- Metasploit oferă o mulțime de tipuri diferite
- Dacă nu există în Metasploit ce aveți nevoie – scrieți de mână
- Cele mai comune tipuri de payload-uri:
  - Bind – ascultă după conexiuni pe un port predefinit
  - Reverse – se conectează pe un host:port predefinit
  - Exec – lansează în execuție o aplicație
  - Download și exec – download fișier din locație predefinită + Exec
  - Loadlibrary – încarcă un DLL
  - Adduser – adaugă un nou utilizator

Tipuri de payload-uri (VI)

- Unele payload-uri prezintă diferite variațiuni:
  - Bind: bind_hidden_ipknock_tcp, bind_tcp_rc4, bind_tcp_uuid, etc.
  - Reverse: reverse_tcp_allports, reverse_tcp_dns, reverse_tcp_rc4, etc.
- Unele payload-uri pot fi livrate în forme diferite:
  - Meterpreter
  - DllInject
  - Staged
  - VNC
- Uneori, dorim să scriem un payload custom, din diferite motive
  - Nu există în Metasploit, vrem să evităm detecții, etc.

Codificarea unui payload

- În funcție de vulnerabilitate și exploit, payload-urile trebuie codificate pentru:
  - A evita caractere nedorite
  - A evita detecția unor antiviruși
- Codificarea se aplică payload-urilor binare, și presupune de obicei criptarea lor
- Există codificatoare pentru: x86/x64, MIPS, Sparc, PPC

Tehnici de exploatare: RCE

- După descoperirea unui serviciu vulnerabil, urmărim exploatarea acestuia
- Trebuie să stabilim ce exploit și ce payload vom utiliza
- Tipul de payload depinde de:
  - Ce vrem să obținem
  - Sistemul vizat
- Uneori, dorim să evităm detecții de antiviruși sau alerte de firewall
  - Prin urmare, vom ajusta tipul payload-ului utilizat
- În principiu, orice tip de payload ne poate ajuta să ne îndeplinim scopul
  - ... oricare ar fi acesta
- Vom considera în principal vulnerabilități de tip RCE

Tehnici de exploatare: RCE (II)

- Bind payload
  - Pornește un listener pe un port predefinit
  - Ne vom putea conecta la acel port și vom obține un shell
  - Dezavantaj: firewall, port filtrat, port deja utilizat, etc.

```
msf payload(meterpreter_reverse_tcp) > use payload/windows/shell_bind_tcp
msf payload(shell_bind_tcp) > show options

Module options (payload/windows/shell_bind_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, process, none)
   LPORT     4444             yes       The listen port
   RHOST                      no        The target address

msf payload(shell_bind_tcp) >
```

Tehnici de exploatare: RCE (III)

- Reverse payload:
  - Se va conecta la un host:port predefinit și va oferi un shell
  - Dezavantaj: firewall

```
msf payload(shell_reverse_tcp) > show options

Module options (payload/windows/shell_reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  none             yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     127.0.0.1        yes       The listen address
   LPORT     80               yes       The listen port

msf payload(shell_reverse_tcp) > _
```

Tehnici de exploatare: RCE (IV)

- Uneori, utilizăm un payload de bind/reverse care nu pare a funcționa
  - Din cauza unor reguli de firewall, de exemplu
- Cum aflăm dacă exploit-ul funcționează într-adevăr?
  - Exemplu: utilizăm un payload de tip exec care face ping la mașina noastră
- În general, există porturi accesibile de pe orice mașină
  - HTTP, HTTPS, FTP, etc.
  - Este recomandat ca payload-ul de reverse să se conecteze pe un astfel de port

Tehnici de exploatare: RCE (V)

- Dacă sistemul are un serviciu de RDP/SSH activ, putem utiliza un payload de tip adduser și apoi să ne conectăm pe RDP/SSH
  - Dacă serviciul de RDP/SSH nu este activ, îl putem activa
    - Dacă avem drepturi de root/admin
- Dacă sistemul are un server web, putem să adăugăm un backoor de tip PHP care interpretează comenzi
  - Fișierul va fi localizat în directorul din care se servesc paginile web
  - Accesând-ul, putem trimite comenzi sistemului
- Dacă sistemul are un server FTP, putem adăuga un nou user cu acces la întregul FS
  - Uneori nu este posibil, dacă nu avem drepturi depline și serverul rulează ca root

Tehnici de exploatare: RCE (VI)

- În cazul unui RCE prezent în servicii web, există mai multe variabile
  - Există numeroare metode de exploatare a RCE-urilor web
  - Depinde de ce engine este folosit de către server
  - Depinde de vulnerabilitate
- Exemplu: Local File Inclusion (LFI)
  - Dacă putem forța scrierea de date pe disk într-o locație cunoscută, este trivial
  - De obicei, putem injecta comenzi/scripturi în loguri (de exemplu, logul Apache)
  - Se pot accesa fișiere sensibile (passwd, shadow, SAM hives)
- Exemplu: Remote File Inclusion (RFI)
  - Este trivial de exploatat: includem un fișier sursă de la noi de pe server

Tehnici de exploatare: RCE (VII)

- Exemplu: credențiale slabe/default într-o aplicație web
  - Poate se pot modifica fișiere sursă existente, se pot uploada plugin-uri, etc.
  - Se poate obține acces la sistem direct ca root/ca un user mai puțin privilegiat
- Exemplu: SQLi
  - xp_cmd_shell

Tehnici de exploatare: non-RCE

- Vulnerabilități de tip NON-RCE pot fi considerate toate vulnerabilitățile care nu conferă execuție arbitrară de cod în mod direct:
  - XSS, CSRF, SQLi, ID, weak/default credentials, etc.
  - O mare parte din vulnerabilități nu sunt reprezentate de RCE
- Dacă nu putem executa cod pe victimă, este mai dificil să o compromitem (oare?)
- Totul depinde de vulnerabilitate și aptitudinile atacatorului și de ce dorim să obținem
- Uneori, acestea pot fi totuși exploatate pentru a compromite victima
- Chiar dacă nu pot duce la compromiterea victimei, tot pot fi utile

Tehnici de exploatare: non-RCE (II)

- Exemplu: SQLi:
  - Obținut informații din baza de date
  - Adăugat înregistrări în baza de date
- Exemplu: XSS
  - Obținem cookie/sessionid de la victime
- Exemplu: CSRF
  - Putem modifica credențiale, adăuga useri noi, etc.
- Exemplu: directory traversal
  - Obținut passwd/shadow, SAM, fișiere confidențiale

Meterpreter

- Payload care oferă un mediu complet de lucru pe mașina compromisă
- Meterpreter există în multiple forme:
  - Binar: x86/x64
  - Script: PHP, Python
- Pe diferite sisteme de operare:
  - Android, Linux, Windows
- Și folosind diferite tehnici:
  - reverse_http, reverse_tcp, bind_tcp

Meterpreter (II)

- Principalul dezavantaj:
  - Este detectat de antivirușsi/IDS/IPS/etc
- Principalul avantaj:
  - O mulțime de comenzi utile și ușurința de a lucra cu el
  - Compatibilitate cu majoritatea exploit-urilor

SHA256:	883e1c96f567a97005fc727f09dfc40227b1c914263af4137442d33bc4ac7e51
File name:	meter.exe
Detection ratio:	35 / 56
Analysis date:	2016-11-25 13:10:40 UTC (1 minute ago)

Meterpreter (III)

- Se pot rula module adiționale de Metasploit
  - run modul
- Se poate crea un shell
  - shell
- Se pot rula comenzi uzuale
  - ps, ls, cat, pwd, cwd, cd, etc.
- Se pot citi hash-urile NT/LM
  - hashdump
- Uneori, se poate face privilege escalation
  - getsystem

Meterpreter (IV)

- Screen-shots
  - screengrab
- Migrare în alte procese
  - migrate
- Pornirea camerelor web:
  - webcam_snap
- Upload/download de fișiere

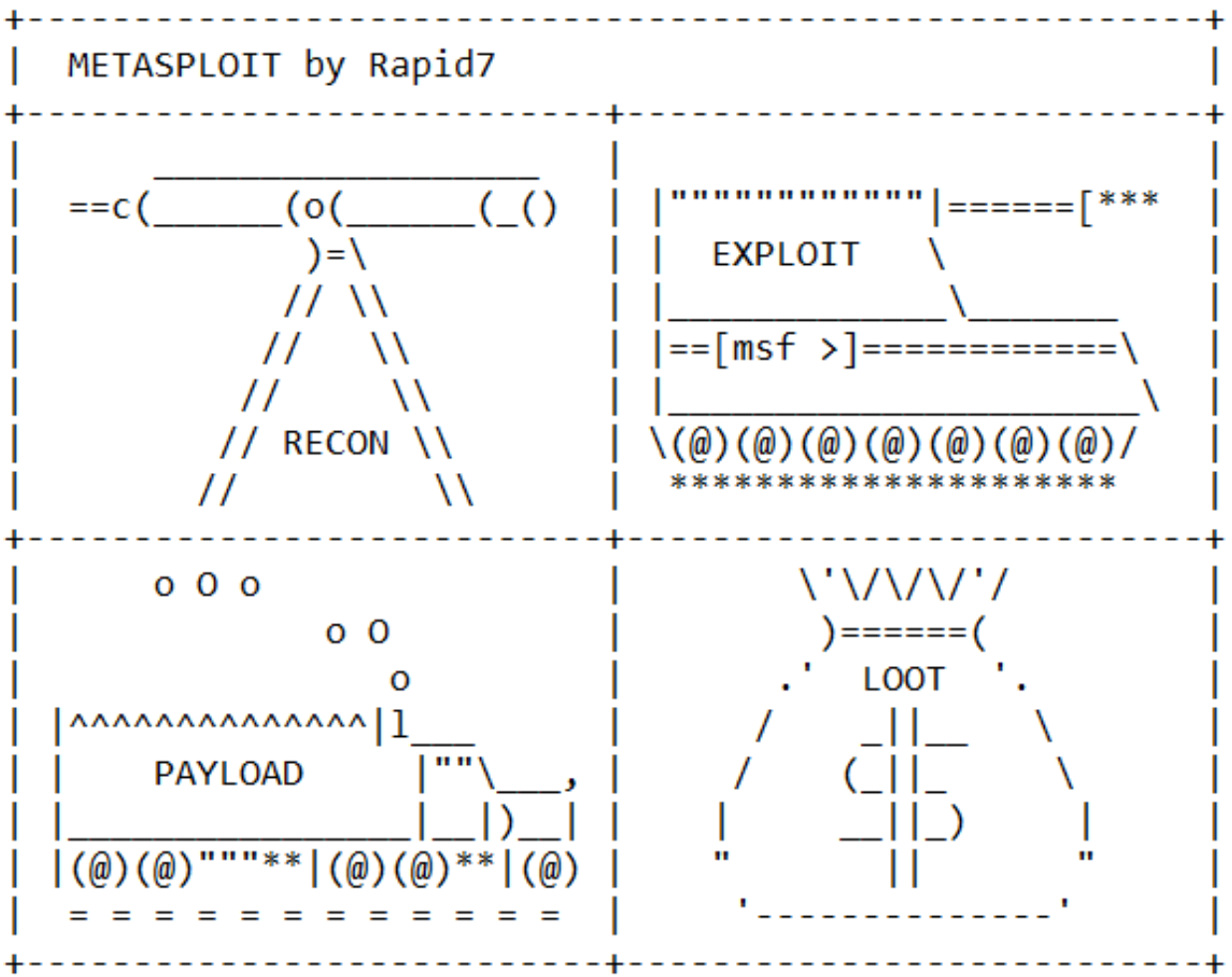
Lateral movement / pivotare

- Lateral movement = compromiterea altor sisteme din rețea
- De cele mai multe ori, un sistem expus în exterior (internet) are multiple interfețe de rețea
- O dată compromis un sistem, există posibilitatea folosirii acestuia pe post de pivot pentru lateral movement
- O dată compromis un astfel de sistem, putem începe compromiterea altor sisteme din rețeaua respectivă
  - Sistemul curent devine pivot
- Dacă s-a obținut acces în interiorul organizației, totul devine mult mai simplu

Metodologie (IV)

- Persistența accesului
  - Instalare de backdoors
  - Adăugarea de useri noi
  - Obținerea de hash-uri/parole pentru acces ulterior
- Enumerare post-atac
  - Informații confidențiale, hash-uri, documente, etc.
- House-keeping
  - Rootkits pentru ascunderea de componente „instalate” pe sistemul atacat
  - Curățarea urmelor
  - Curățare loguri de acces, history, etc.
  - Ștergere fișiere/componente instalate

Metodologie (V)



Unelte

- Sistem de operare
  - Kali Linux
    - Distribuție bazată pe Debian
    - Conține o mulțime de unelte specifice
- Unelte de scanare/exploatare în masă (nerecomandate)
  - OpenVAS, CoreImpact, SAINT, Nessus, NeXpose
- Unelte de bază
  - Metasploit, nmap

Unelte (II)

- Obținere de informații
  - Netdiscover, nmap, Maltego, etc.
- Analiză vulnerabilități
  - Nmap, Golismero, OpenVAS, etc.
- Analiză aplicații web
  - Burpsuite, WebScarab, etc.
- Analiză baze de date
  - SQLMap, SQLNinja, etc.
- Password attacks
  - john, hashcat, rainbowcrack, etc.

Unelte (III)

- Pentru rețele wireless
  - aircrack-ng, etc.
- Reverse engineering
  - OllyDbg, NASM, clang, apktool, etc..
- Exploatare
  - Metasploit, SQLMap, armitage, etc.
- Sniffing & spoofing
  - Wireshark, ettercap, responder, etc.
- Post-exploatare
  - ProxyChains, bdfproxy, etc.

Unelte (IV)

- Forensics
  - Volatility, binwalk, etc.
- Raportare
  - Keepnote

Raportare

- Procedura de pentesting se încheie cu un raport
- Raportul trebuie să includă:
  - Fiecare sistem vulnerabil
  - Fiecare vulnerabilitate identificată
  - Metodologia/pașii de exploatare
  - Analiza riscului
  - Soluții
- Raportul este elaborat de pentester și înmânat clientului
- Informația trebuie comunicată clar și eficient
  - Clientul nu e prea educat în domeniul securității...

Bibliografie

[1] The Basics of Hacking and Penetration Testing, Second Edition: Ethical Hacking and Penetration Testing Made Easy (Engebretson, Patrick – 2013 – Sygress)
[2] Metasploit: The Penetration Tester's Guide (Kennedy, David – 2011 – No Stach Press)
[3] Web Penetration Testing with Kali Linux (Muniz, Joseph – 2013 – Packt Publishing)
[4] Hacking Exposed – Network Security Secrets Exposed (McClure, Stuart – 2012 – McGrawHill) (7th ed)
[5] https://www.sans.org/reading-room/whitepapers/testing/writing-penetration-testing-report-33343