Virtual Machines Description

Virtual Security Lab

by Alexis SEMREN and Steven MARIEN

History	Who	Version	When
Creation	AGGS3/SM861	1.0	19/08/2015

1 - Virtual Machines

1 - 1 Privilege Escalation - ubuntu kernel vulnerability

Virtual Machine Name	Escalate - Boot2Root
System version	Ubuntu 6.10 with a kernel 2.6 version
User	**toto** —> **NApregUstu6r+dRe**
Description	This machine is vulnerable due to a kernel vulnerability. SSH password is provided to the student.
Open Port	SSH: **22**
Process	There are many exploits available for this kernel version, quick look on Google says it all.
Goal	Escalation privilege - user has to escalate to root - basic linux enumeration - to apply existing exploits to a vulnerable machine
Flag	/

1 - 2 Privilege Escalation - Misconfiguration NFS

Virtual Machine Name	Escalate - Boot2Root
System version	Debian 7
User	**root** —> **5_tUBuv7GU7putha** **toto** —> **jemuspuGEspE7e+e**
Description	This machine has a misconfigured NFS daemon that allows anyone to write a root file in the filesystem. SSH password is provided to the student.
Open Port	SSH: **22** NFS: **2049**
Process	- noticing NFS daemon running - checking NFS daemon configuration files - no_root_squash option activated - mounting the NFS in a personal machine with root access - writing a code to provide a root shell - executing this code by normal user on the machine
Goal	Escalation privilege - user has to escalate to root - basic linux enumeration - to know how to exploit vulnerable configurations
Flag	/

1 - 3 Infrastructure Penetration Testing - Active Self-defense

Virtual Machine Name	Active Self-defense
System version	Debian 7
User	**root** —> **root** **toto** —> **toto4vm**
Description	This machine sets up honeypot ports and ban the IP if something looks suspicisous. Artillery is used for that purpose.
Open Port	SSH: **22** + Honeypot ports
Process	- stealth scanning - grabbing banners - getting detected + banned - lucky shot to SSH port with fun password "root" gives access
Goal	Understanding that defensive mechanisms are powerful
Flag	In /home/toto/flag —> **thisisthetoken**

1 - 4 Infrastructure Penetration Testing - Self-Monitoring

Virtual Machine Name	Self-monitoring
System version	Debian 7
User	**root** —> **root** **toto** —> **toto4VM**
Description	This machines sets up honeypot ports and alerts if something hits the honeypot. Artillery is used for that purpose.
Open Port	SSH: **22** + Honeypot ports
Process	- port scanning - grabbing banners - only SSH port looks 'normal' - bruteforcing SSH port (fun password 'root') - flag contains the logs and all alerts that have been triggered
Goal	Understanding monitoring systems. SOC exist and are pretty powerful. Almost anything can be detected.
Flag	In /home/toto/flag **1234567890**

1 - 5 Infrastructure Penetration Testing - FTP

Virtual Machine Name	Ftp
System version	Debian 7

User	**root** —> **D=sP!HUdu5utrE2=** **ftpuser** —> **rEsPefR@5#Beruna**
Description	It is a ProFTPD (version 3.3c) with a remote shell vulnerability.
Open Port	SSH: **NO** FTP: **21**
Process	Use **Kali** to detect the vulnerability and use **metasploit** to create and send a **payload**.
Flag	Flag in /root/pass.txt —> **w?a3a@tUBEnap2ba27!p-@a47e8ASU=hunUf**

1 - 6 Web - Web exercises

Virtual Machine Name	Web
System version	Debian 7
User	**root** —> **6eqa6EDuPrAh$4Ej8t#a** MySQL: **root** —> **-@a47e8ASU=hunUf** **jack** —> **XUtrusa-uce8r*pe** **wordvuln** —> **w2a7Ra7ema_RAk=P**

Description	This machine is focusing web vulnerabilities such as:
	- Sql Injection (two levels and database in Polish to make it harder):
	  - in the authentication form, 0 defense.
	  - authentication form defended, vulnerability is in the product page id=x.
	- XSS (four levels, no tokens):
	  - filtering quote/double quotes (Comment Form)
	  - filtering “<” and “>”
	  - 0 defense
	  - filtering “script” word
	- Cookie (one level):
	  - stored information
	- Url access restriction (one level):
	  - changing url's values
	- Form upload (one level):
	  - No protection during the file upload
	- Obfuscation (two levels):
	  - The password is in the main.min.js file in plaintext
	  - String.fromCharCode(x,x,x,x,x,) in the main2.min.js file

Open Port	SSH: **NO** Apache2: **80** MySQL: **3306**

Process	**SQL Injection:** - simple **' OR '1'='1' -- '** - The second one encourages to use tools (**Sqlmap?)** in order to find the card number, it is a blind SQL vulnerability in a polish database. **XSS:** - Payload **<script>alert(1);</script>** works on this exercise - Payload **" onmouseover="'alert('gg')"** works on this exercise - Payload **<script>alert('Hello');</script>** works on this exercise - Payload ** Funny ** works on this exercise **Cookie:** - **Changing** the cookie value from "visitor" to "**admin**" **URL:** - **Changing** URL parameter from "right" to "**administrator**" **Form_upload:** - This exercise is a form upload without any verification, the user can upload a **payload** to have a shell access or just display some message. Once the file is upload, the user need to find it in the **files/** directory. **Obfuscation:** - The password can be found directly in the **main.min.js** file - The user need to find the code (**String.fromCharCode**(x,x,x,x,x,)) and execute it.
Goal	- Learn the basis of web penetration testing - OWASP top ten vulnerabilities
Flag	SQL Injection: - level 1 —> **@aZu_u@a3rugur?nazaC** - level 2 —> **44092030588494949** Cookie —> **@aZu_u@a3rugur?nazaC** Url —> **4EQupr=Chut#R3xE?a** Form_upload: **/lab_web/exercises/form_upload/files/** —> **f-c7eC2Aqup3Yeyuk@d** Obfuscation: - level 1 —> **admin:kAc*aWre3aSta#eJ=wru** - level 2 —> **admin:dRUFRusuthudr&3=Cru-**

1 - 7 Password attack - Brute Force

Virtual Machine Name	Web **(Same machine as 1 - 6)**
Description	It is a web page with an **authentication form**
Process	Use a password cracker like **Hydra** to find the password
Flag	**admin:!canon123**

1 - 8 Infrastructure Penetration Testing - Wordpress

Virtual Machine Name	Web **(Same machine as 1 - 6)**
Wordpress User	**jack4wordpress —> spU8ecr@vAzugaj4**
Description	It is a **Wordpress** blog with **4 vulnerable plugins**: • wp-FileManager 1.3.0 - **File Download Vulnerability** • DB Backup Plugin - **Arbitrary File Download** • Accept Signups 0.1 - **XSS** • Easy Slideshow Plugin - **CRSF**
Process	Use of **WPSCAN** to find the plugins and follow the links to see how to **exploit** the vulnerabilities.
Flag	There are no flag

2 - Files

2 - 1 Password attacks - Hash

Filename	Hashes
Description	Linux hashes from a /etc/shadow file 6th type: SHA512
Process	Any password attack tool
Goal	- Understanding where and how are stored passwords on Linux systems - Understanding weak algorithms and why Linux uses salts.
Flag	- joshua - soccer - phantom - wizard

2 - 2 Reverse Engineering - Buffer overflows

Filename	overflow1, overflow2, overflow3
More info	Compiled with **gcc overflow.c -o overflow -fno-stack-protector**
Description	To be able to compile without **overflow protection**: → echo 0 > /proc/sys/kernel/randomize_va_space There are three different buffer overflow exercises. The code is available in the web application in the Reverse engineering section.
Process	**overflow1**: Just smash the stack --> “cat /dev/urandom	./overflow1” **overflow2**: - Create an unique string and send it to crash the program - Find wich part of the unique string smashes eip - Using this offset to smash eip with win() address (target address) **overflow3**: - Objdump the binary in order to find the address of the targeted if condition - Open edb and at the above address, change the value to 0
Goal	- introducing linux memory management system - introducing debugging tools and C programs - learn how to smash the stack and redirect execution flow
Flag	overflow1 —> **c6Uxa422E!ETruw*EceC** overflow2 —> **4reNuCha2As_bRat2brU** overflow3 —> **4up4yUd@druzu6hE=uba**