Operating Systems Security – Assignment 1

Version 1.1 – 2016/2017
Due Date: 18 Nov 2016 (23:59 CET)

1 Manage custom PAM authentication rules

Pre-requisite: Setup virtualized Kali Linux operating system

- Download, install and configure the Kali Linux VirtualBox 64-bit image (not the 'light' version), see website[1]
- Add a few (test) users to the system. Use the `adduser` command to do this.

Objectives

Play around with the Pluggable Authentication Modules (PAM) in the Kali Linux system.

a) For each of the PAM control values (required, requisite, optional, sufficient), give an example of a PAM rule using it, which is actually useful in some context. Explain the context where the rule should be used and what the rule accomplishes.

b) Create the text file `/tmp/users` and specify on each line a valid username; you have to specify at least one user.
In this exercise, limit yourself to only adjust the rules in the sshd PAM module configuration file (`/etc/pam.d/sshd`).
Use the `pam_listfile` module[2] and try to achieve the following sshd login configurations for the users listed in `/tmp/users`:

   i) Disable remote password logins for the specified users.
   ii) Disable remote public key logins for specified users.
   iii) Bypass authentication and allow remote user logins without a valid password or authorized public key.

   Hand in your solution for each of the previous rules and point out why you applied the corresponding control value. If you were not able to compose a PAM directive that restricts/allows one or more of the previous rules, explain briefly why you think this is not possible.

   Note: For some background knowledge about PAM, please refer to the following websites [3][4]

   Testing ssh logins: To test `ssh` logins on your Kali VM, you will need to start the `ssh` daemon: `systemctl start ssh`. Afterwards, you can run `ssh localhost` to connect to the local machine.

2 Write your own PAM module

You are required to write a basic custom PAM module which asks a user 1 out of 5 questions randomly and the user is required to provide the correct answer. You are free to be as creative as you like with these 5 questions.
We advise you to execute **`sudo apt-get install libpam0g-dev`** and test your module using **`su`** (and not **`login`** or **`ssh`**).

[1] https://www.offensive-security.com/kali-linux-vmware-virtualbox-image-download/
[2] http://www.linux-pam.org/Linux-PAM-html/sag-pam_listfile.html
[3] http://www.linux-pam.org/Linux-PAM-html/Linux-PAM_SAG.html
[4] https://www.netbsd.org/docs/guide/en/chap-pam.html

You need to hand the source code of the module together with a Makefile to build it and a config file `/etc/pam.d/su` that uses the module for authentication.
Note: For additional background knowledge about PAM, please refer to the following websites [5][6][7]

3 Become familiar with the Metasploit Framework

Pre-requisite: Download and setup a vulnerable system

- Download and setup Metasploitable. We have provided a VirtualBox OVA appliance for easy importing[8]. Metasploitable provides an intentionally vulnerable Linux virtual machine. Note, **never** expose this VM to an untrusted network (e.g. set it up behind a router / NAT / Host-only configuration). Import Metasploitable and Kali (64-bit). Make sure both VMs are on the same network, figure out their IP addresses and check that you are able to `ping` the Metasploitable VM from your Kali VM.

- Download in Kali Linux the Nessus "home" edition from Tenable network security[9] and register for an activation code on their website[10]. Install the Nessus Debian package:
  `# dpkg -i Nessus-6.9.0-debian6_amd64.deb`
  And start Nessus with:
  `# /etc/init.d/nessusd start`
  Start Firefox, (or download another browser) and open the local Nessus web-interface:
  `https://localhost:8834`

- If you want you can try out some of the tools included with Kali, such as `Sparta`, or you can install `OpenVAS`[11].

Objectives

Define a "Basic Network Scan" and run it against the Metasploitable VM. You will find in this system at least a couple of problems which are identified by Nessus as "critical". Some of those problems are trivial to exploit (like logging into a service with username "admin" and password "admin"). Other problems require some more effort, but are efficiently exploitable with Metasploit. Some practical examples using Metasploit are explained on the following websites[12][13][14]

a) Exploit the vulnerability that Nessus discovered in *vsftpd* with Metasploit. Give a brief summary what actions you performed and which (additional) sources you have used to exploit the system.
b) Exploit vulnerability CVE-2010-2075 in *Unreal IRCD*. Again, give a summary of the actions you performed and which sources you used to perform the exploit. Of course, if you want to play more with Metasploit, feel free to keep exploiting more vulnerabilities.
c) (optional) Exploit the vulnerability in *vsftpd* manually using standard Linux tools.

4 Buffer-overflow attack

This exercise is meant to serve as a refresher and serves as a prerequisite to better understand the lecture on *Memory* (Lecture 2). Some might already have done this exercise for the "Hacking in C" course, but you are still expected to hand in a solution.

[5] http://www.linux-pam.org/Linux-PAM-html/Linux-PAM_SAG.html
[6] http://www.rkeene.org/projects/info/wiki/222
[7] http://www.wpollock.com/AUnix2/PAM-Help.htm
[8] https://rded.nl/ossec/metasploitable.ova
[9] http://www.tenable.com/products/nessus/select-your-operating-system
[10] http://www.tenable.com/register, we suggest you use https://mailinator.com
[11] https://www.kali.org/penetration-testing/openvas-vulnerability-scanning/
[12] http://www.securitytube.net/video/5489
[13] http://www.securitytube.net/video/6432
[14] https://community.rapid7.com/docs/DOC-1875

- Download the code from https://www.cs.ru.nl/~vmoonsamy/teaching/ossec2016/a1_ex4.zip.
- Compile the downloaded code:
  ```
  # make
  ```
- Disable ASLR (as root):
  ```
  # echo 0 > /proc/sys/kernel/randomize_va_space
  ```
- To execute the vulnerable code:
  ```
  # ./vulnserv
  ```

The exercise can be found here:
http://www.cs.ru.nl/~erikpoll/sws1/exercises/assignment6.pdf. You need only look at the second exercise. The supplied code is the code that ran on `hackme.cs.ru.nl`, so replace any `nc` commands by just running the program directly.