Project 2: Exploitation

Miguel Archundia

Jacobo Sanchez

Derek Hernandez

Salvador Romero Mondragon

Section I (Introduction)

**Summary**:

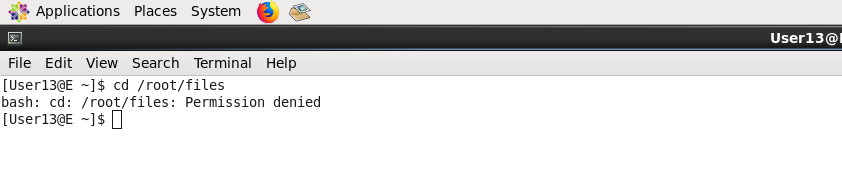
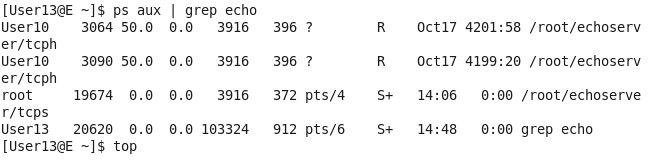
The main point of Project two was to not only setup the network and root permissions but test whether the custom attack scripts will work in the task of retrieving the files from a remote computer. This task presents the strategy of manipulating stack memory. After the attack is successful we then test what kind of defenses are used to protect memory manipulation.

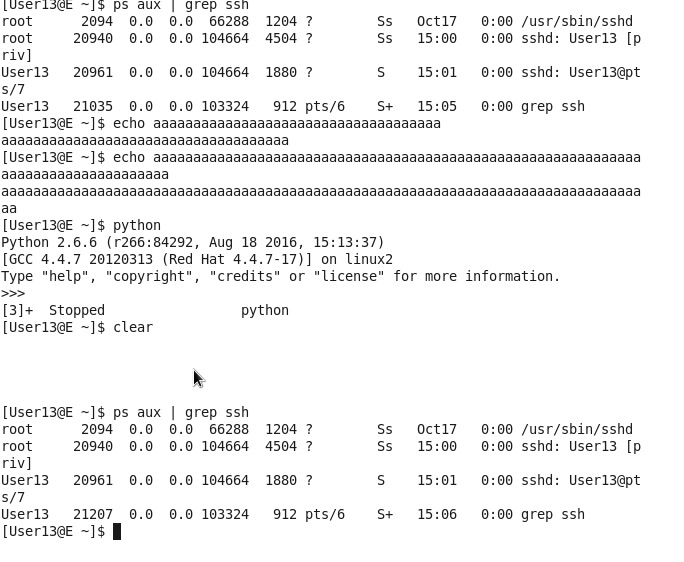
**Responsibilities**:

Project: Task I and II were performed by Jacob and Miguel with support from Salvador. Task III and IV were performed by Jacob and Miguel with support from Salvador and Derek.

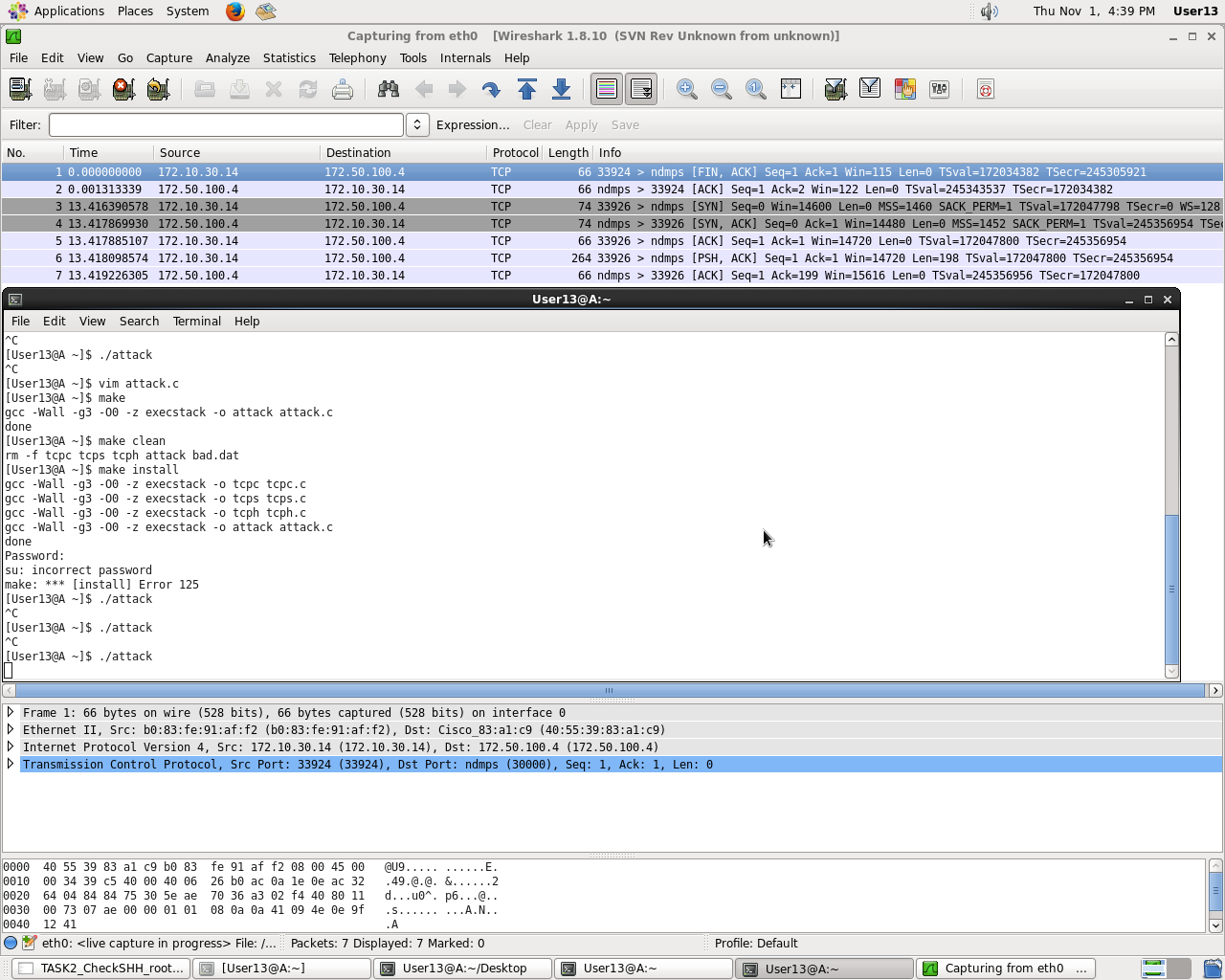
Report: Sections I-II were done by Miguel, Jacob edited the final report and wrote section IV with a suggestion from Derek, section I was done by Derek, sections II-III were done by Salvador.

Section II (Task II)

1. .
   1. Not able to access /root/files without root password
   2. The maximum allowable size of the data allowable in an echo command is 65,507 bytes. Anything more than 8 bytes.
   3. Showing user ID for echo services in E.2
   4. Showing user ID that is running SSH services on computer E.2.

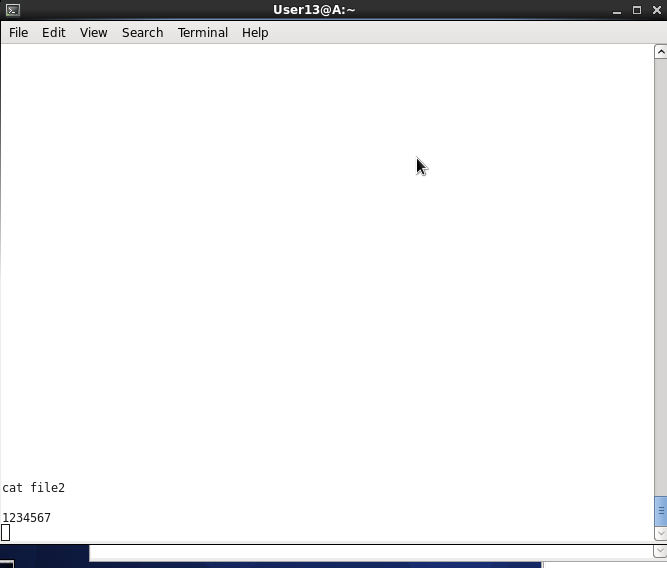


Section III (Task III)

1. .
   1. Showing that echo services can be exploited by the provided shell code.(shown in the image below)
   2. Showing the exploited packets captured in Computer E.2 (shown in the image below)
   3. Report how you retrieve the files from computer E.2 to E.B. Give steps in detail

Solution provided by Jacob. After hacking into the server, target files were copied into the /var/www/html directory. The files were then downloaded to the attacking server using the wget function, i.e. attacked server ip/filename.

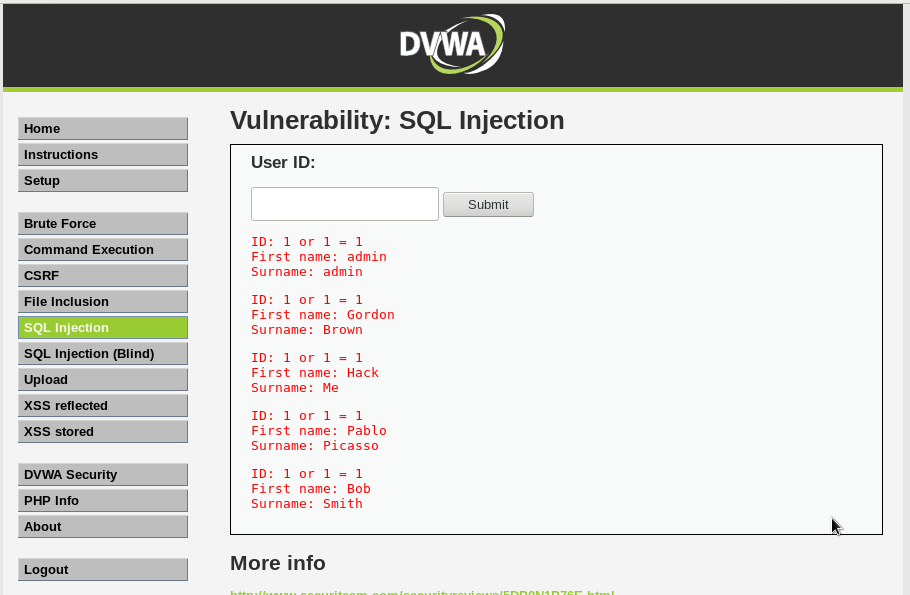
* 1. .



* 1. The statement we used to display all of the user ID is

(1 or 1 = 1)

* 1. Showing the screenshot of the web page that show all user ID’s, first names, and last names.



Section IV

a) Discuss the reason that randomization can defeat the attack.

Randomization does not defeat attacks but minimizes the chances of script attack success by randomizing target addresses. This makes attacks that rely on using specific memory locations more difficult.

b) Assume only the low 16 bits of the stack address is randomized. What is the probability that an exploiting packet can compromise the server? Assume an attacker can send 10 exploiting packets every second. How long can the attacker compromise the server?

Assuming isolated guessing where address space is re-randomized after each attack attempt, the probability of server compromisation is g = 1 - ( 1 -2^-N)^a where a is 0 <= a => g = 1 - ( 1 - 2^16) ^10 = .001526\*100% = .1526%

https://en.wikipedia.org/wiki/Address\_space\_layout\_randomization

The attacker can compromise the server until the server OS reacts to program crashes due to attack failures.

c) Discuss the reason that exec-shield can defeat the attack.

Exec-shield defeats the attack because the script changes the permissions of executables in the memory.

d) Discuss if exec-shield prevents stack overflow. If not, what attack can be achieved?

Exec shield in combination with position independent executables randomization patch make heap and format string exploits almost impossible. An application that is not fully compatible is still vulnerable, however.