**Lab 3 : TCP Attack Lab**

Due : August 6, 2024, 11:59 PM

**Task 1 (45 Points): SYN Flooding Attack**

* Were you able to launch a successful attack with synflood.py? Provide a screenshot. (Output of task 1.1)

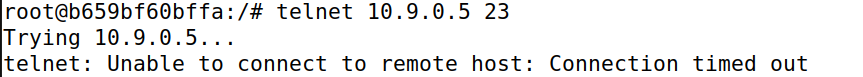
**The attack failed since we were able to log in, this might be because python runs a slow synflood attack.**

Изображение выглядит как текст, снимок экрана, Шрифт, документ

Автоматически созданное описание

* Were you able to launch a successful attack with synflood.c? Provide a screenshot.

**This attack seemed to have succeeded, since we were not able to connect to the victim’s machine using telnet**



* Were you able to launch attack after enabling SYN cookie mechanism? Justify your answer. Provide a screenshot.

**After enabling the SYN cookie mechanism**, our **attacks both synflood.c and synflood.py have failed. Even when running synflood.py file we were still able to log in, even though it took a much longer time than synflood.c . The reason why the attack failed is that the machine detected that it was under a SYN flooding attack and filtered out the false IP requests.**

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание

**Task 2 (15 Points): TCP RST Attack on telnet Connection**

* Were you able to break the TCP connection between a user1 and the victim server? How did you launch the TCP RST attack? Provide evidence with screenshots.

**Yes, in order to break the TCP connection, we first had to establish the connection between the user1 and the victim server. After launching the python file, we tried entering a message in the user’s terminal, and it prompted us with the Connection Failed message. Thus, meaning that the connection was broken.**

Изображение выглядит как текст, Шрифт, чек

Автоматически созданное описание Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Изображение выглядит как текст, Шрифт, белый

Автоматически созданное описание

**Task 3 (20 Points): TCP Session Hijacking**

* Were you able to hijack the TCP session? How did you launch the session hijack? Provide evidence with screenshots.

**Yes, after editing the** **sessionhijack.py, we have established a connection between user1 and the victim. Next, we launched a second terminal for the attacker in order to capture the attack result, in other words, read a file from the victim’s machine. Using the nc -lnv 9090 command in the second attacker terminal, we start listening for our attack results. When our user enters any text during his connection to the victim, our attack outputs the content of the victim’s text file, causing the user terminal to freeze.**

Изображение выглядит как текст, снимок экрана, Шрифт, число

Автоматически созданное описание **Code**

**First attacker terminal output**

Изображение выглядит как текст, снимок экрана, меню, число

Автоматически созданное описание

**Second attacker’s terminal output with the listener turned on Изображение выглядит как текст, Шрифт, снимок экрана, белый

Автоматически созданное описание**

**Task 4 (20 Points): Creating Reverse Shell Using TCP Session Hijacking**

* Were you able get access to the shell after launching the attack? What mechanism or parameters did you change to create a reverse shell? Explain and provide screenshot.

**Yes, we have successfully accessed the shell after launching the attack. First we had to modify our sessionhijack.py a bit in order to send the right command into the victim’s machine. Next, we have done the same procedure of the user1 connecting to the victim, and launching a second attacker terminal in order to listen for our attack. However, as the result, our second attacker terminal gained access to the victim’s terminal.**

**The code is the same for this task, except the command stored in our data variable**



**Our second attack terminal gains access to the victim’s terminal**

Изображение выглядит как текст, Шрифт, снимок экрана, белый

Автоматически созданное описание