

Lab 5: SSL Strip Attack

Introduction:

This report will explain, in detail, the steps to a man in the middle SSL Strip Attack using an ARP spoof.

The steps to the attack are as follows: The victim does not know the website runs over https or does not care about this detail. The victim types www.website.com and the browser default to http. The attacker intercepts this traffic and forces the victim to do an http interaction with the attacker. The attacker poses as the victim and does the https transaction with the server on the victim's behalf: yet does an http transaction with the victim's browser, therefore a "man in the middle attack." The browser wont ask for the certificate since it is not aware that the page you are asking for is an https protected page. This attack proceeds with no certificate warning.

Preparation Steps:

1. Creating two VMs in Oracle VM Virtual Box Manager/Selecting a NAT Network:

Network: First I began by cloning my current VM and naming it "Kali 2." (I later added the attribute "[ATTACKER]" to the Kali2 name for better clarification) I then created a NatNetwork in the virtual box manager settings. Finally I connected both VMs to a NAT network. See **Figure 1.1-1.2** below.

Figure 1.1

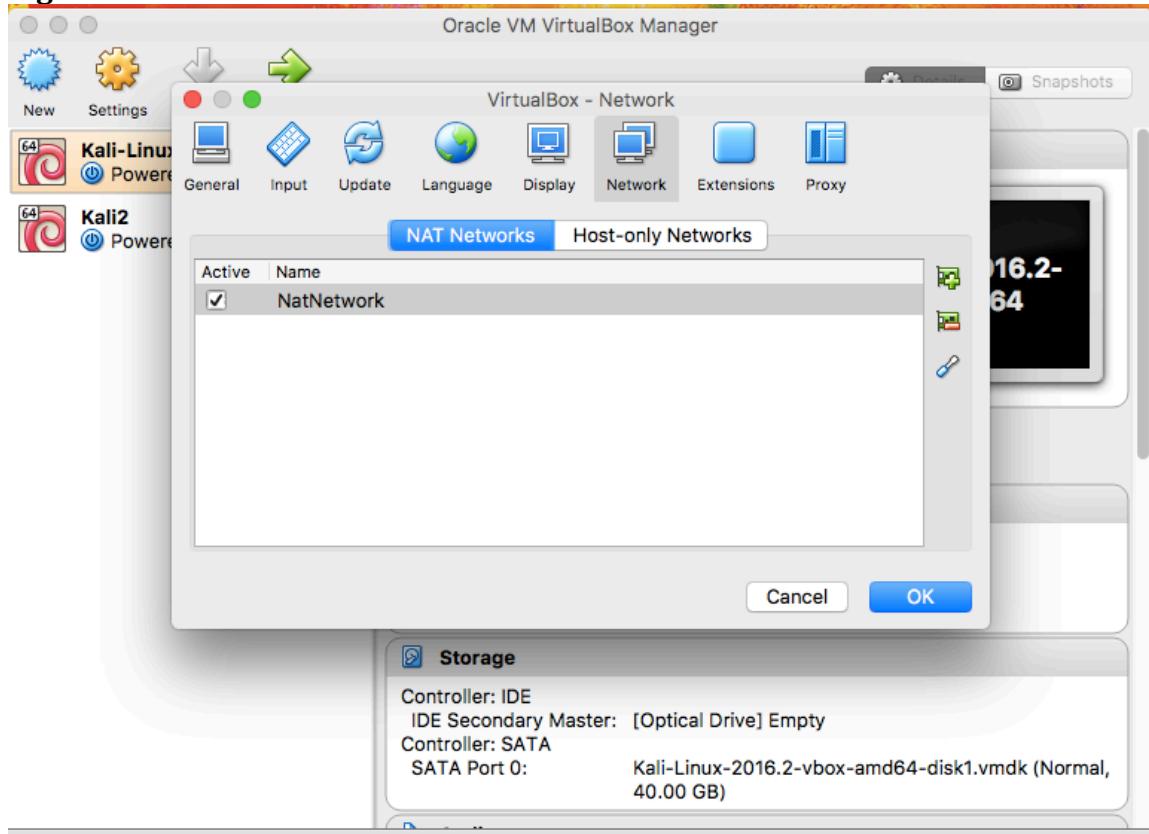
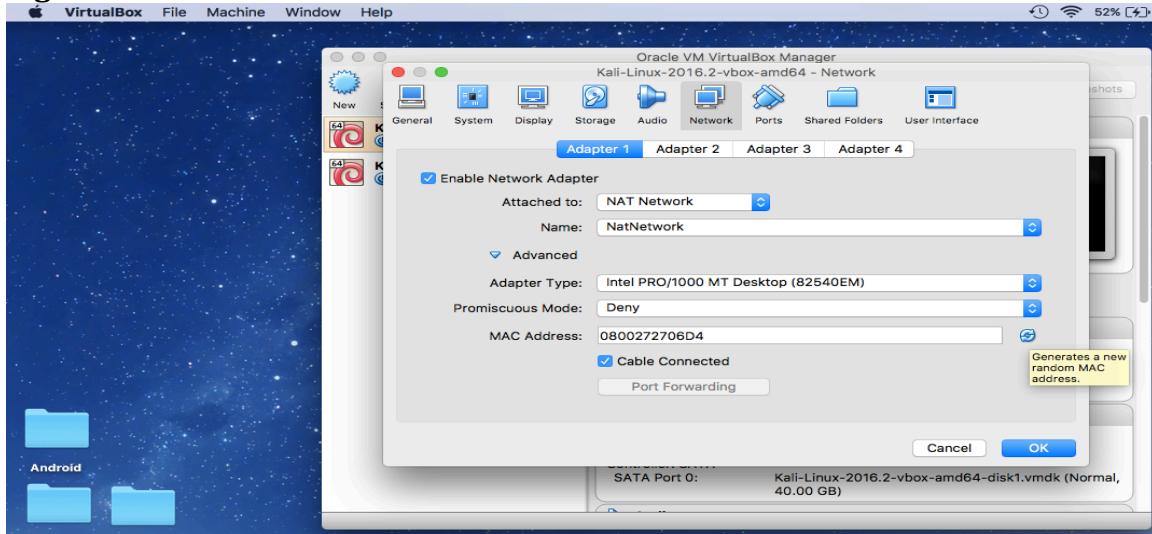


Figure 1.2



2. **Verifying VM Communication:** The last preparation step is making sure the VMs can communicate with one another and that they are both connected to the internet. First I identified the IPs of both VMs, see **Figure 2.1** below. Using the `ip addr show` without the interface name, is how I identify the network interface name. After identifying the IPs I then verified that the machines could communicate with each other by using the ping command. See **Figure 2.2** below.

Figure 2.1

```
root@kali:~# ip addr show eth0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
group default qlen 1000
    link/ether 08:00:27:2a:11:ae brd ff:ff:ff:ff:ff:ff
        inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic eth0
            valid lft 867sec preferred_lft 867sec
            inet6 fe80::a00:27ff:fe2a:11ae/64 scope link
                valid lft forever preferred_lft forever
root@kali:~#
```



```
root@kali:~# ip addr show eth0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP
group default qlen 1000
    link/ether 08:00:27:4d:d4:c8 brd ff:ff:ff:ff:ff:ff
        inet 10.0.2.4/24 brd 10.0.2.255 scope global dynamic eth0
            valid lft 1167sec preferred_lft 1167sec
            inet6 fe80::a00:27ff:fe4d:d4c8/64 scope link
                valid lft forever preferred_lft forever
root@kali:~#
```

Figure 2.2

```
Kali-Linux-2016.2-vbox-amd64 [Running]
root@kali:~/Desktop/Lab5
File Edit View Search Terminal Help
root@kali:~/Desktop/Lab5# ping -c 1 10.0.2.4
PING 10.0.2.4 (10.0.2.4) 56(84) bytes of data.
64 bytes from 10.0.2.4: icmp_seq=1 ttl=64 time=3.16 ms
...
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 3.168/3.168/3.168/0.000 ms
root@kali:~/Desktop/Lab5# 

Kali2_[ATTACKER] [Running]
root@kali:~/Desktop/Lab5
File Edit View Search Terminal Help
root@kali:~/Desktop/Lab5# ping -c 1 10.0.2.15
PING 10.0.2.15 (10.0.2.15) 56(84) bytes of data.
64 bytes from 10.0.2.15: icmp_seq=1 ttl=64 time=2.33 ms
...
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 2.339/2.339/2.339/0.000 ms
root@kali:~/Desktop/Lab5# 
```

Attacker VM Steps:

3. **Forwarding Packets:** The first step to preparing the attacker VM is configuring Kali to forward incoming packets. I also verified that the correct file was created in the desired location using the `echo "1" > /proc/sys/net/ipv4/ip_forward` command. See **Figure 3.1** below.

Figure 3.1

```
Kali2_[ATTACKER] [Running]
root@kali: /proc/sys/net/ipv4
File Edit View Search Terminal Help
root@kali:~/Desktop/Lab5# cd /proc/sys/net/ipv4
root@kali:/proc/sys/net/ipv4# ls
conf
tcp_frto
tcp_fwmark_accept
tcp_invalid_ratelimit
tcp_keepalive_intvl
tcp_keepalive_probes
tcp_keepalive_time
tcp_l3mdev_accept
tcp_limit_output_bytes
tcp_low_latency
tcp_max_orphans
tcp_max_reordering
tcp_max_syn_backlog
tcp_max_tw_buckets
tcp_mem
tcp_min_rtt_wlen
tcp_min_tso_segs
tcp_moderate_rcvbuf
tcp_mtu_probing
tcp_no_metrics_save
tcp_notsent_lowat
tcp_orphan_retries
tcp_pacing_ca_ratio
tcp_pacing_ss_ratio
tcp_probe_interval
tcp_probe_threshold
tcp_recovery
ip_forward
ip_forward_use_pmtu
ipfrag_high_thresh
ipfrag_low_thresh
ipfrag_max_dist
ipfrag_secret_interval
```

4. Gateway Address: The next step is using the *netstat* command and identifying the network gateway address. This will be the same for both VMs. See **Figure 4.1** below.

Figure 4.1

Kali2_[ATTACKER] [Running]

Applications ▾ Places ▾ Terminal ▾ Thu 19:10 1

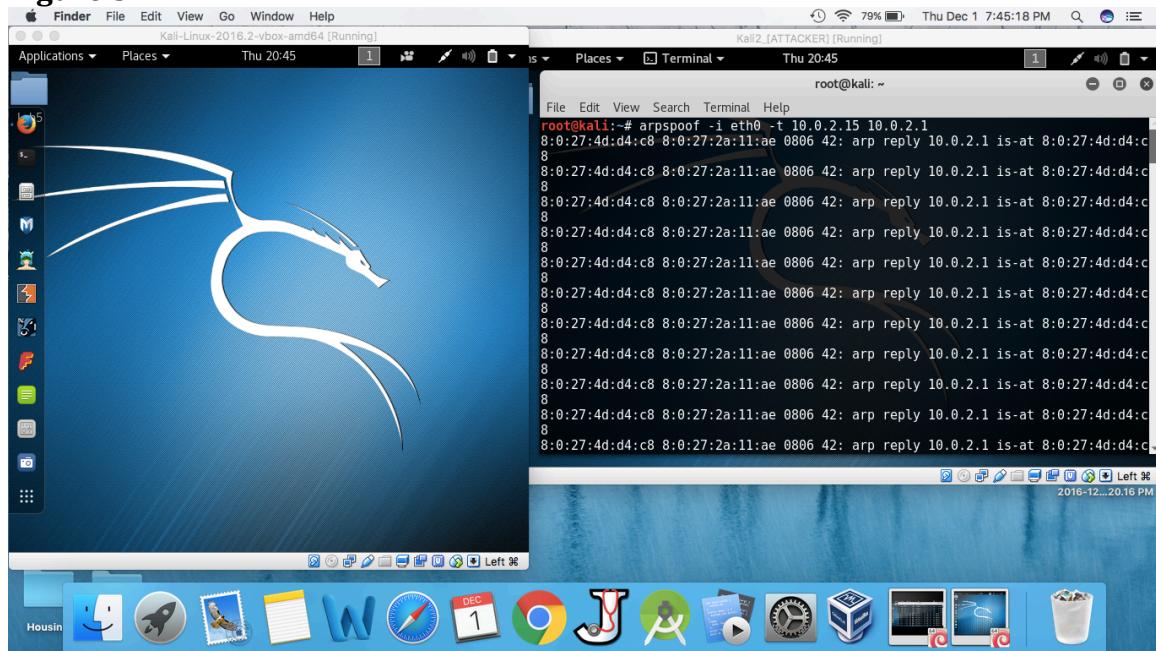
root@kali: ~/Desktop/Lab5

File Edit View Search Terminal Help

```
root@kali:~/Desktop/Lab5# netstat -nr
Kernel IP routing table
Destination      Gateway          Genmask         Flags   MSS Window irtt Iface
0.0.0.0          10.0.2.1        0.0.0.0        UG        0 0          0 eth0
10.0.2.0         0.0.0.0         255.255.255.0  U         0 0          0 eth0
root@kali:~/Desktop/Lab5#
```

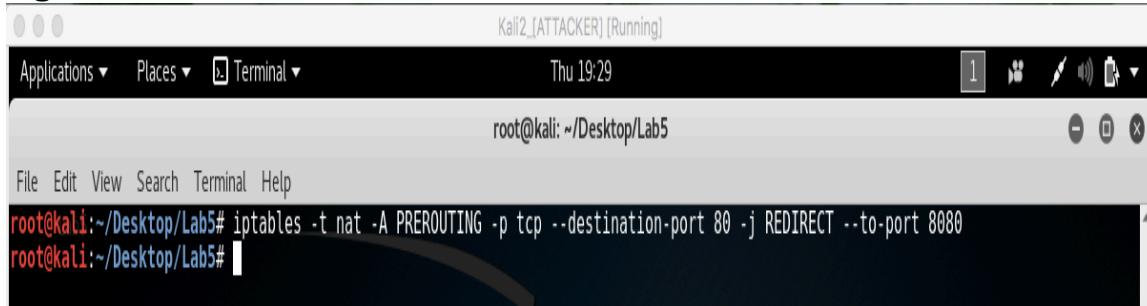
5. **ARPSpoof:** The first step to starting the attack is redirecting the traffic meant for the gateway using the arpspoof tool. See **Figure 5.1** below. After running the arpspoof, I left this terminal window open and opened a new terminal window to continue.

Figure 5.1



- 6. Firewall Rules:** In the new terminal window I set up the firewall rules on the attack machine to redirect traffic received on port 80 to port 8080. See **Figure 6.1** below.

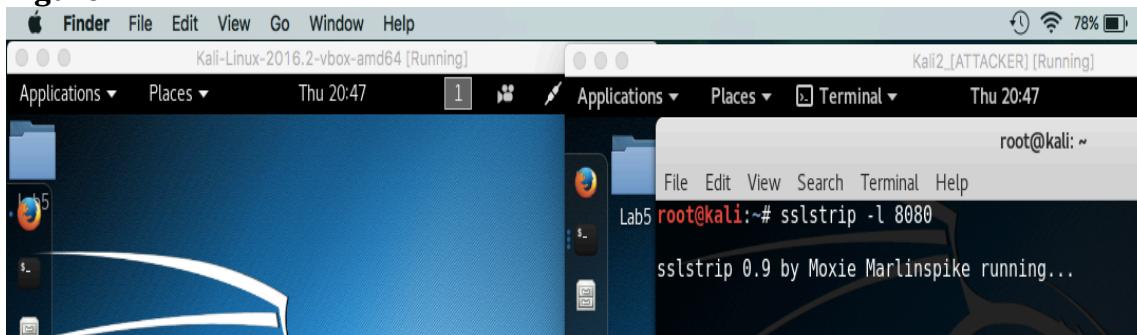
Figure 6.1



Kali2_[ATTACKER] [Running]
root@kali:~/Desktop/Lab5# iptables -t nat -A PREROUTING -p tcp --destination-port 80 -j REDIRECT --to-port 8080

- 7. Run SSLStrip:** Still working in the new terminal window along with the arpspoof window still running, I began the SSLstrip attack. See **Figure 7.1** below.

Figure 7.1



- 8. SSLStrip.log:** I left the SSLStrip and the arpspoof terminal window's open and opened a new terminal window. Now in this new terminal window I viewed the log file produced by the SSLStrip attack. At this moment I switched to the Victim VM and attempted navigating to securityisfun.com/CS4331. I noticed that as the victim the attacker only successfully completes the attack if the victim does not enter https into the search bar. Although this only works if the victim begins the domain name with `http://www...` or just `www`. If the victim starts the domain name with "securityisfun" with no "www," the browser will default the settings to https and the attack will be unsuccessful. I also noticed when the victim does not use the "`http://`" the attacker's strip attack throws an "unhandled" error and the website does not direct the victim to the correct page. With the error the attack is still successful. When the victim uses the "`http://`" the attack and the website behave as expected. See **Figure 8.1-8.5**

Figure 8.1: Shows the victim navigating to the website with out using http://

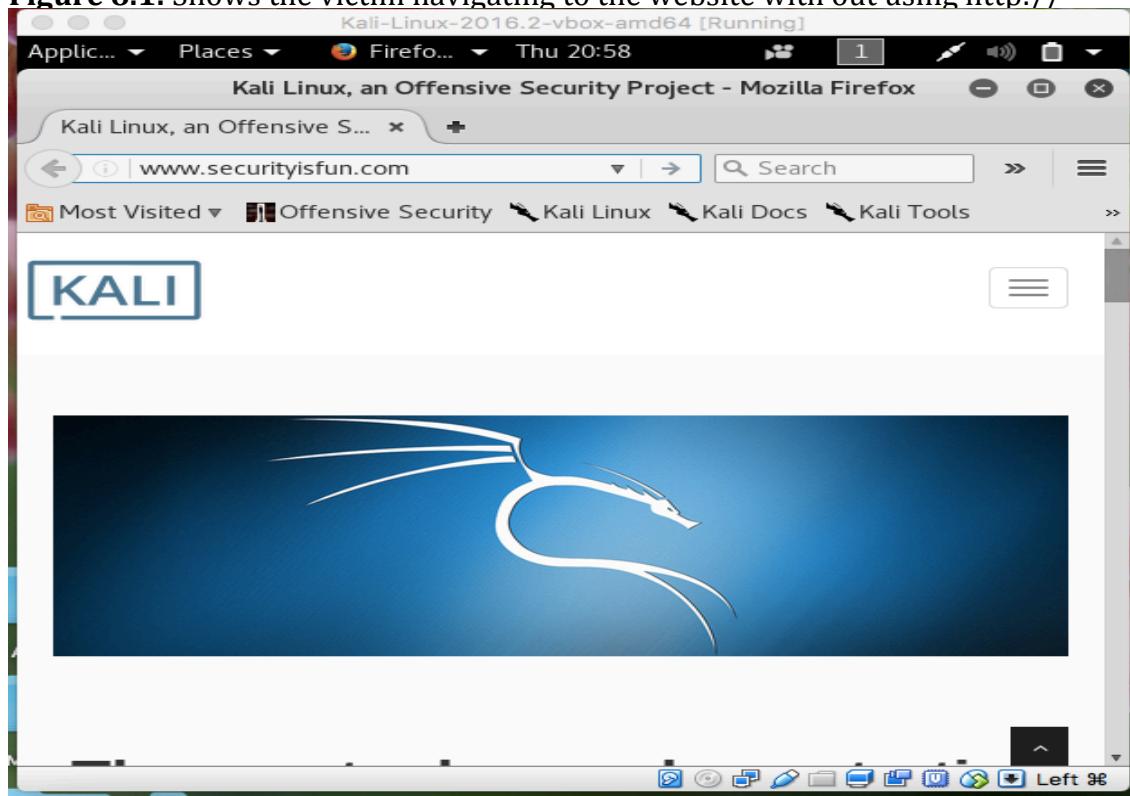


Figure 8.2: Shows the victim navigating to the website using http://

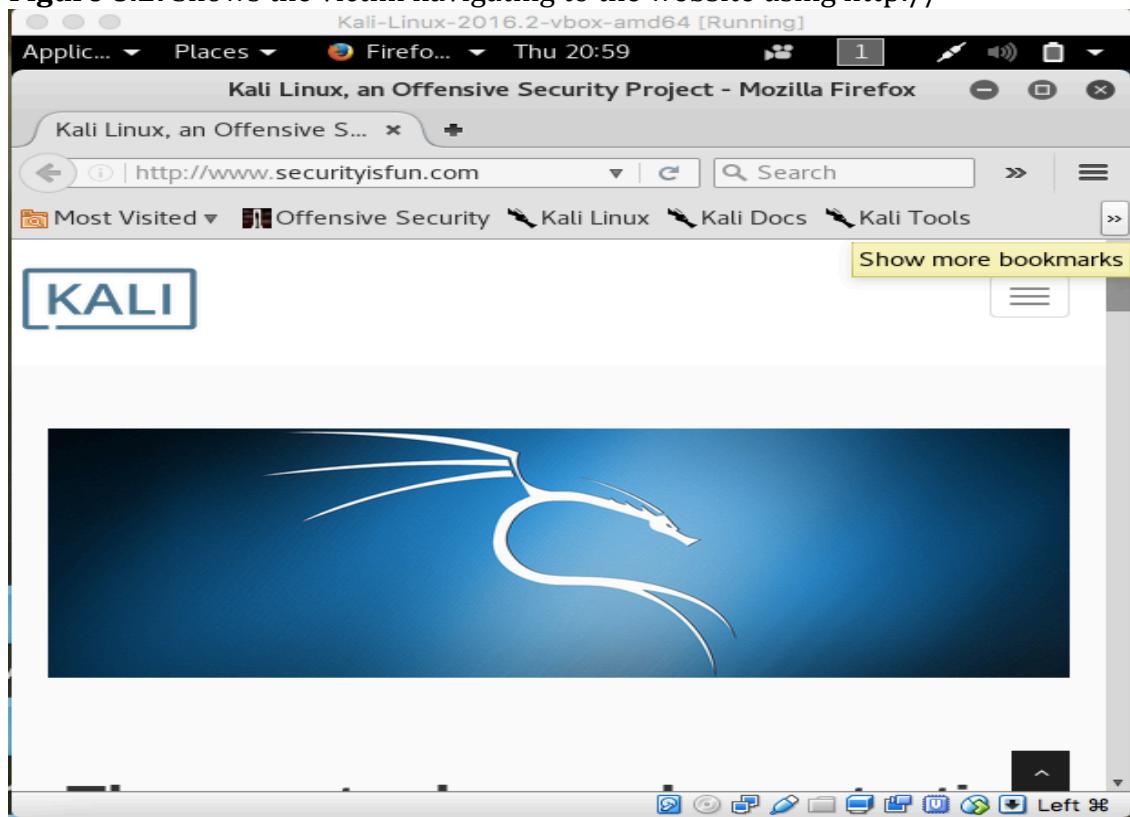


Figure 8.3: Shows the victim entering their credentials

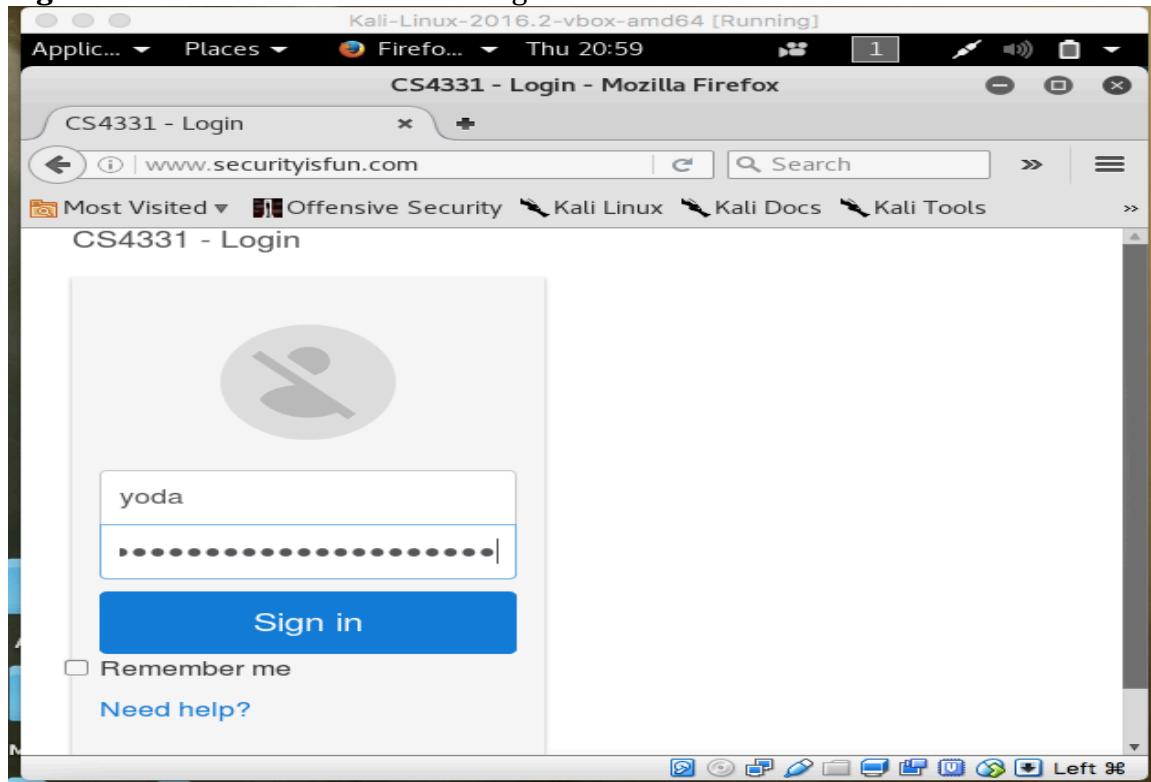


Figure 8.4: Shows the SSLstrip.log (when victim does not use http://) after credentials were submitted to the website

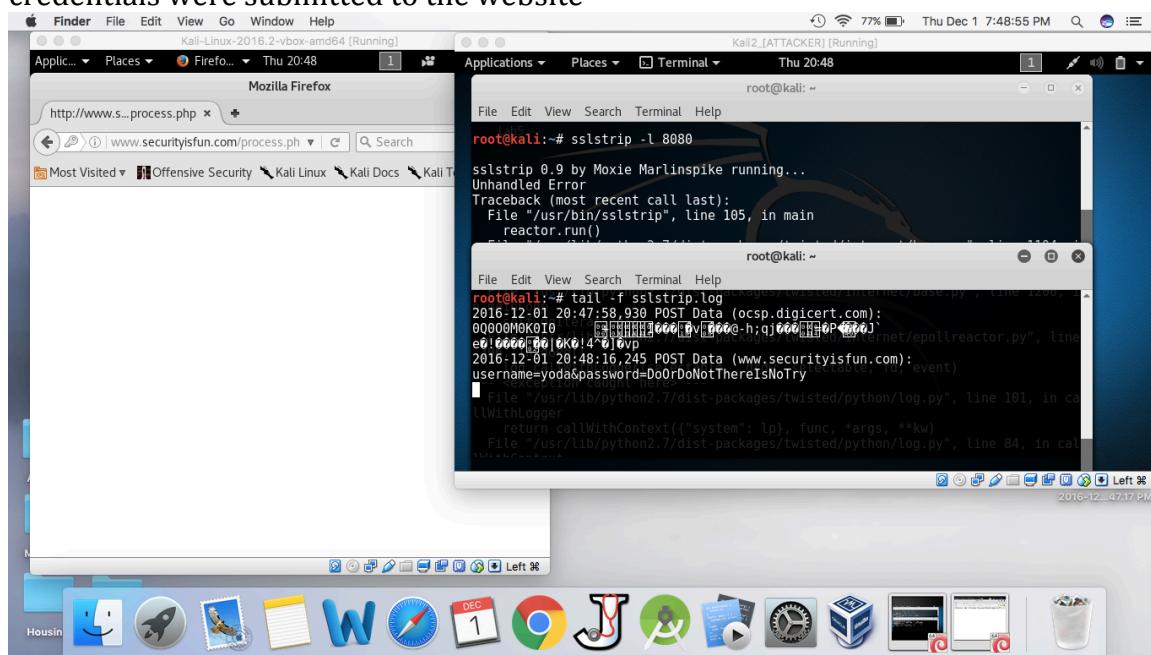
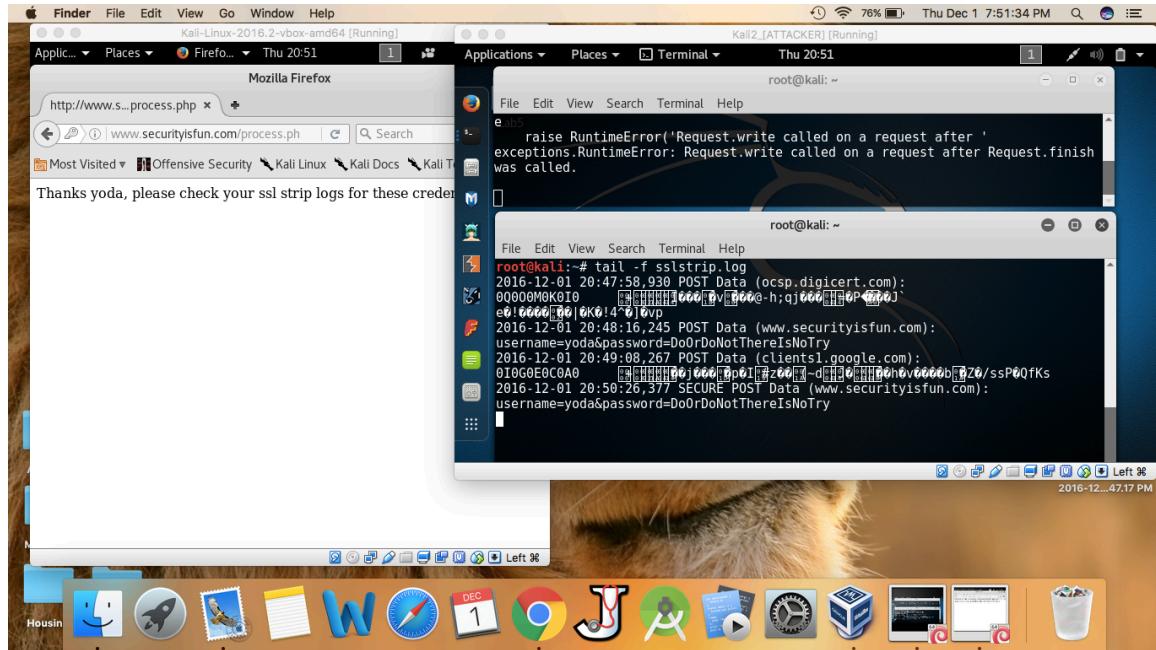


Figure 8.5: Shows the SSLstrip.log (when victim uses <http://>) after credentials were submitted to the website



Conclusion:

In conclusion the attack is successful because the attacker is posing as the victim and does the https transaction with the server. The browser proceeds without any certificate warnings. The attacker is able to intercept any data the victim enters into the website because of the “man in the middle” interaction with the server. This attack is successful when the victim does not specify the https in the search bar. The attack is unsuccessful when the victim either enters the https or lets the browser determine the presets by beginning the domain name with “securityisfun” leaving off the “www.” (The browser will add the https:// automatically) This is due to the HTTP Strict Transport Security, which can protect the victim against this attack. This protocol allows the server to declare to the browser “I must be accessed via HTTPS for the next x amount of years.” If this protocol is implemented, even if the victim types www.webiste.com the browser will know that it has to initiate the request as an https.

But the protocol has its weakness. If the server is accessed for the first time, then it has not had a chance to tell the browser about the https protocol. This means the attack would be successful during this instance. This is more likely to occur on obscure, rarely accessed websites.

Also when the period of activity of the HSTS policy expires, this may also be a window of opportunity for the adversary.

There are also counter measures to this weakness. The Strict Transport Security preloaded list is preloaded on browsers. Allowing the browser to contain a list of known sites, which use https. Which covers the problem of first accessing the server. But does not scale to cover all websites in the world.

The best way to safe guard against this attack is to type https:// straight into the browser all the time. There is still a non-zero probability of falling to the attack, but as close to zero as you can get.