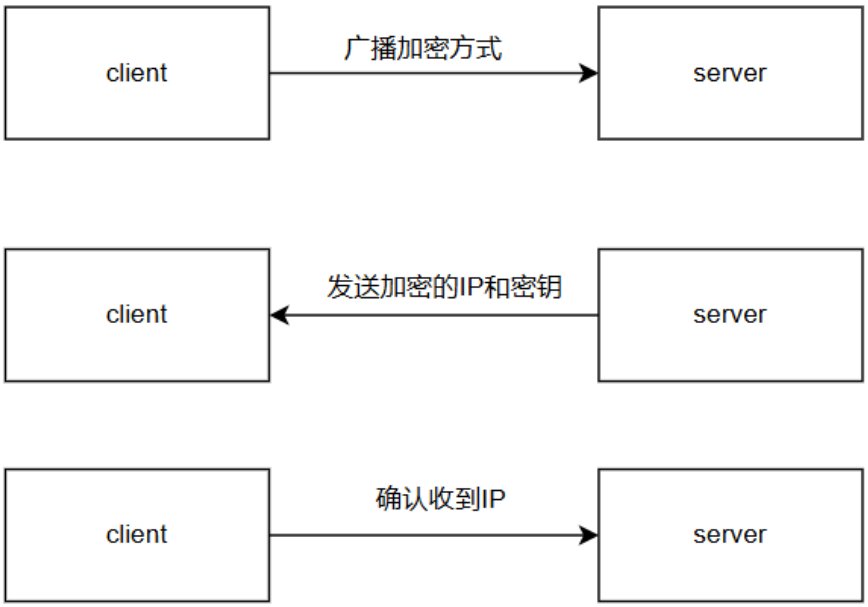
Write up

802.11 帧数据交换如下：



进入 pack，通过 SSID 发现给予的提示，即字符加‘A’（0x41）：

```
630 4.516611128   fe:48:2a:e6:bb:c7   Intel_aa:e3:26   802.11   118 Beacon frame, SN=0, FN=0, Flags=........, BI=100, SSID="plusA"
631 4.517821024   fe:48:2a:e6:bb:c7   Intel_aa:e3:26   802.11   118 Beacon frame, SN=0, FN=0, Flags=........, BI=100, SSID="plusA"
632 4.519506194   fe:48:2a:e6:bb:c7   Intel_aa:e3:26   802.11   118 Beacon frame, SN=0, FN=0, Flags=........, BI=100, SSID="plusA"
633 4.521072795   fe:48:2a:e6:bb:c7   Intel_aa:e3:26   802.11   118 Beacon frame, SN=0, FN=0, Flags=........, BI=100, SSID="plusA"
634 4.529374478   fe:48:2a:e6:bb:c7   Intel_aa:e3:26   802.11   123 Beacon frame, SN=0, FN=0, Flags=........, BI=100, SSID="plusA"
```

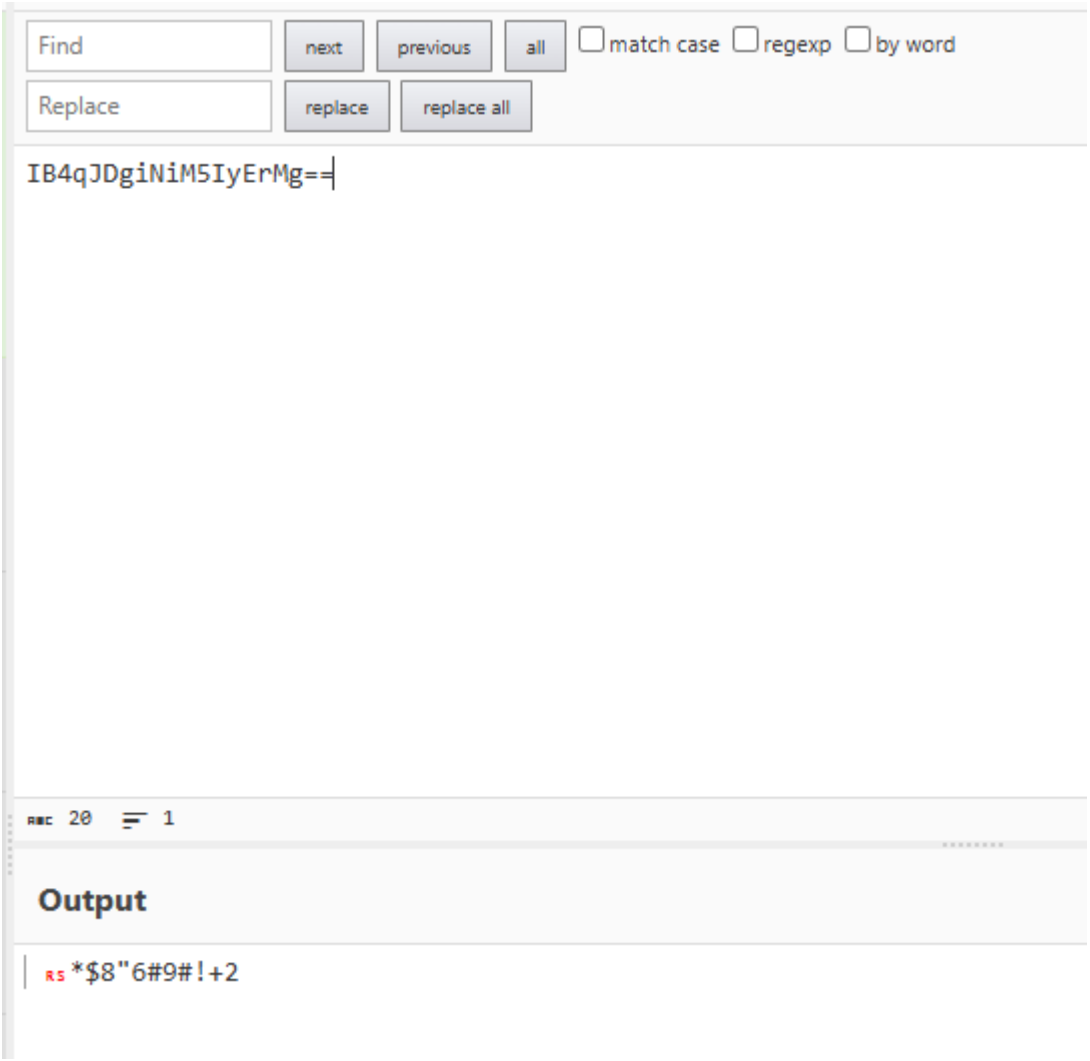
在 plusA 的帧内，有一个标记为 null 的字段：

```
> Extended Capabilities: 0x75 (octet 8)
v Tag: Vendor Specific: (null)
    Tag Number: Vendor Specific (221)
    Tag length: 32
    OUI: 39:73:32
    Vendor Specific OUI Type: 50
    Vendor Specific Data: 32323233353635340049423471 4a4467694e694d35497945724d673d3d
```

该字段存在 base64：

```
0000  00 00 08 00 00 00 00 00  80 00 00 00 50 e0 85 aa   ........ ....P...
0010  e3 26 fe 48 2a e6 bb c7  fe 48 2a e6 bb c7 00 00   .&.H*... .H*.....
0020  00 00 00 00 00 00 00 00  64 00 11 00 00 05 70 6c   ........ d.....pl
0030  75 73 41 01 04 82 84 8b  96 03 01 05 bf 0c fa 79   usA..... .......y
0040  9b 33 fa ff 00 00 00 00  00 00 7f 08 fb eb 83 7a   .3...... .......z
0050  9c 8f 9e 75 dd 20 39 73  32 32 32 32 33 35 36 35   ...u. 9s 22223565
0060  34 00 49 42 34 71 4a 44  67 69 4e 69 4d 35 49 79   4.IB4qJD giNiM5Iy
0070  45 72 4d 67 3d 3d                                 ErMg==
```

把 base64 解开：



用上 ssid 给予的提示，加上'A'：

```
IB4qJDgiNiM5IyErMg==
```

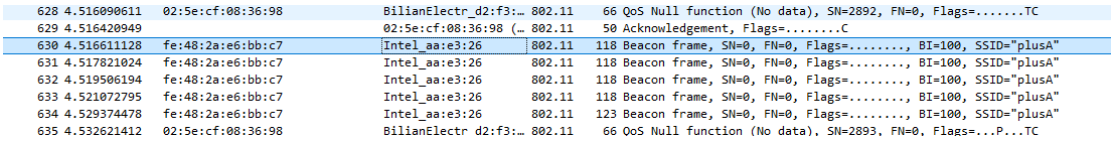
ABC 20 1

Output

```
a_keycwdzdbls
```

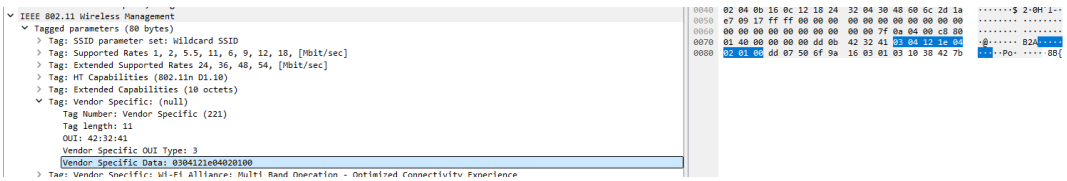
得到一个 key ： “cwdzdbls”

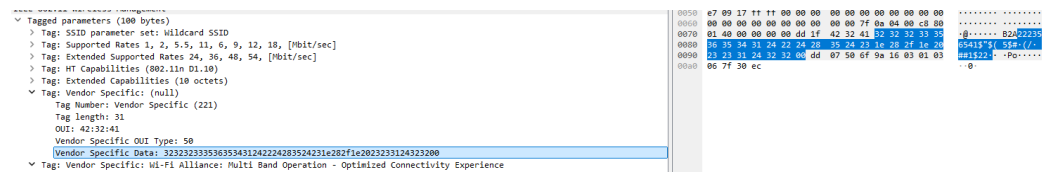
plusA 帧发给指定 mac 地址 intel_aa:e3:26：



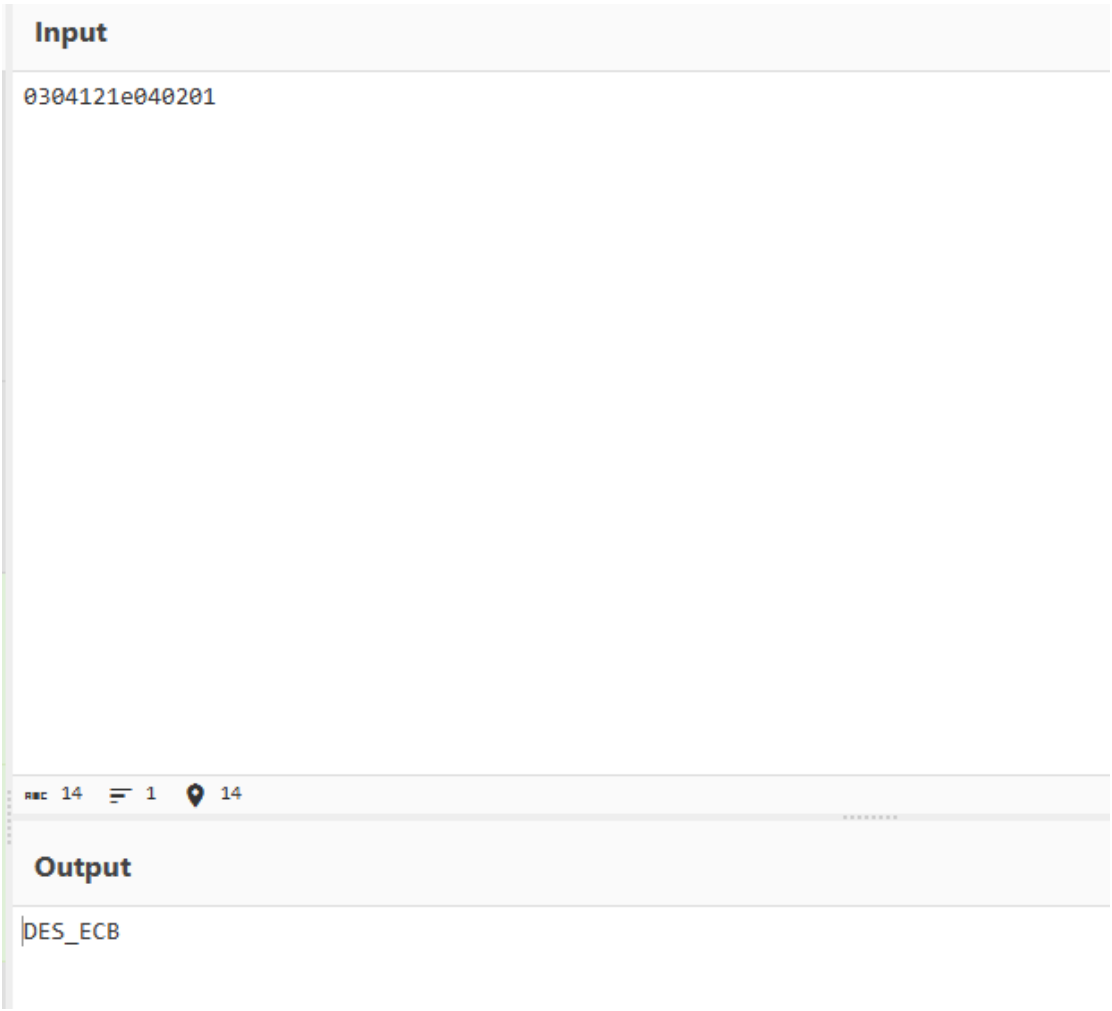
628	4.516090611	02:5e:cf:08:36:98	BilianElectr_d2:f3:…	802.11	66 QoS Null function (No data), SN=2892, FN=0, Flags=.......TC
629	4.516420949		02:5e:cf:08:36:98 (…	802.11	50 Acknowledgement, Flags=........C
630	4.516611128	fe:48:2a:e6:bb:c7	Intel_aa:e3:26	802.11	118 Beacon frame, SN=0, FN=0, Flags=........, BI=100, SSID="plusA"
631	4.517821024	fe:48:2a:e6:bb:c7	Intel_aa:e3:26	802.11	118 Beacon frame, SN=0, FN=0, Flags=........, BI=100, SSID="plusA"
632	4.519506194	fe:48:2a:e6:bb:c7	Intel_aa:e3:26	802.11	118 Beacon frame, SN=0, FN=0, Flags=........, BI=100, SSID="plusA"
633	4.521072795	fe:48:2a:e6:bb:c7	Intel_aa:e3:26	802.11	118 Beacon frame, SN=0, FN=0, Flags=........, BI=100, SSID="plusA"
634	4.529374478	fe:48:2a:e6:bb:c7	Intel_aa:e3:26	802.11	123 Beacon frame, SN=0, FN=0, Flags=........, BI=100, SSID="plusA"
635	4.532621412	02:5e:cf:08:36:98	BilianElectr_d2:f3:…	802.11	66 QoS Null function (No data), SN=2893, FN=0, Flags=...P...TC

查看 intel_aa:e3:26 向外广播的帧。所有帧的 vendor specific 字段皆被标记为 null，有两种不同的 vendor specific 内容：





处理第一种，对 vendor specific 字段数据加’A’，得到加密方式 DES_ECB:



对第二种的 vendor specific 字段数据加'A'，得到信息 intel_aa:e3:26 已收到某个 IP 地址：

Input

```
323232333536353431242224283524231e282f1e20232331243232
```

54 1

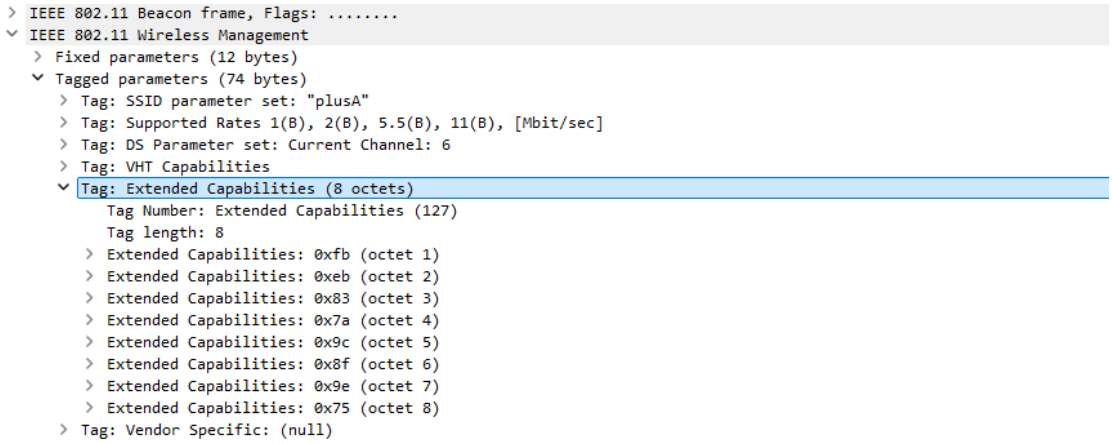
Output

```
ssstvwvureceived_ip_address
```

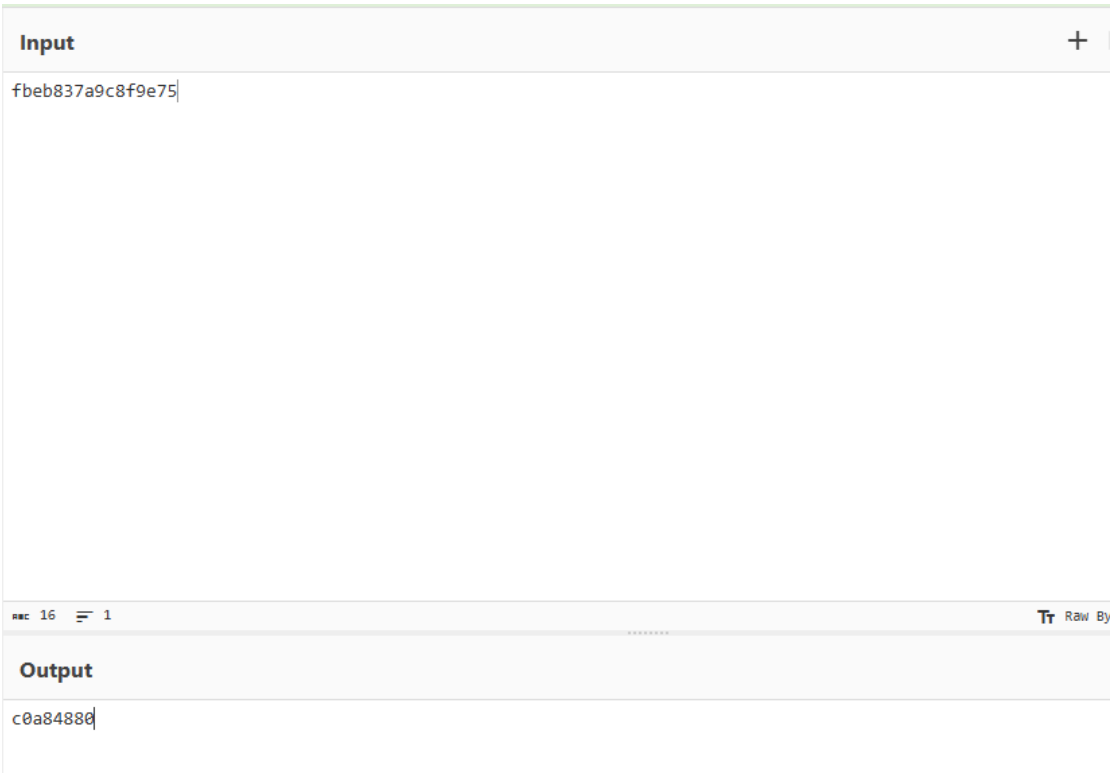
查看帧发送顺序并总结已知信息：

1. intel_aa:e3:26 先向外广播加密方式 DES_ECB。
2. plusA 向 intel_aa:e3:26 发送密钥"cwdzdbls"和其他数据。
3. intel_aa:e3:26 广播收到 IP 地址。

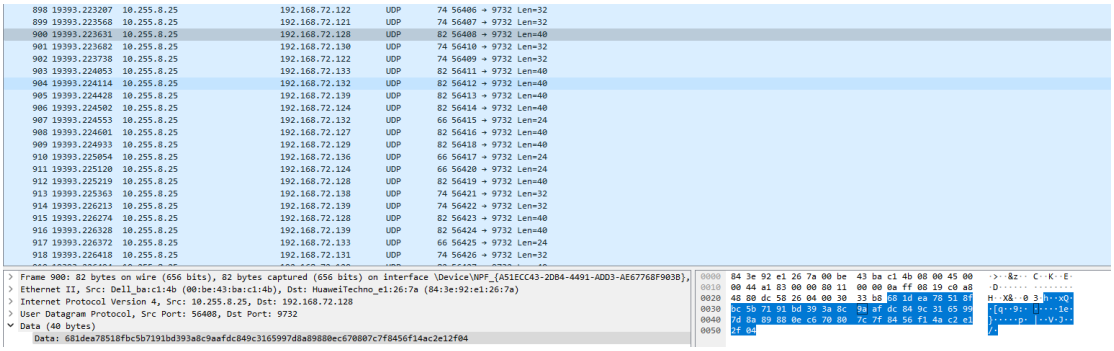
根据 DES_ECB 密钥为八字节，加密输出结果为八字节的倍数，再次分析 plusA 数据包，并发现刚好有一个字段为 8 字节：



对该字段进行 DES_ECB 解密，得到一个 ip 地址，16 进制为 192.168.72.128：



查看下方 udp 包，提取发向 192.168.72.128 的 udp 包 data：



根据提示，对 data 进行 DES_ECB 解密，再用获取到 IP 的 Extended Capabilities 字段的 Tag number（127）进行异或，得到 flag{4nt1y_c3rt_Apt_@n41y51s_gr0up}：

Input

```
681dea78518fbc5b7191bd393a8c9aafdc849c3165997d8a89880ec670807c7f8456f14ac2e12f04
```

80 1

Output

```
flag{4nt1y_c3rt_Apt_@n41y51s_gr0up}
```