

IAP2 协议

IAP2 全称是 Ipod Accessory Protocol 2 协议，是苹果设备和第三方配件之前数据传输、控制的协议。一个 IAP 连接是由一个 IAP 链路和一个或者多个会话组成。其中建立 IAP 链路是 IAP 会话的基础，所有控制和传输都是基于一个链路。

其中：IAP2 协议的包主要有三大类：心跳（握手）包、链路包、会话包

IAP2 心跳（握手）包

IAP2 心跳握手包主要用来判断附件是否支持 IAP2 协议。当设备连接上之后，附件应通过以 1Hz（每秒一次）发送以下字节序列来确认是否存在支持 iAP2 的设备，直到从该设备收到响应：0xFF 0x55 0x02 0x00 0xEE 0x10。如果设备支持 iAP2，则附件将返回到相同的字节序列。

The image shows a Wireshark packet capture of a Bluetooth connection. The first packet (463) is a Bluetooth HCI ACL Packet from the iPhone to the accessory, containing a 19-byte IAP2 heartbeat packet (0xFF 0x55 0x02 0x00 0xEE 0x10). The second packet (484) is a Bluetooth RFCOMM Packet from the accessory to the iPhone, containing a 19-byte response (0xFF 0x55 0x02 0x00 0xEE 0x10). The packets are highlighted in red and blue respectively.

上述就是表示 IAP2 的心跳包，用来判断设备是否支持 IAP2 协议，如果支持的话，也会回复 0xFF 0x55 0x02 0x00 0xEE 0x10 表示支持 IAP2。

IAP2 Link 链路包

iap2 的链路对会话传输数据的时候提供支持，相对 session 会话来说，iap2 链路的建立比 session 要早。链路的连接一般是 iap2 连接的开始，链路提供了多种传输方式在设备和配件之间建立通信。

1. 蓝牙
2. Usb Device/Host Mode
3. 其他传输方式

下面分析过程传输方式均基于蓝牙

IAP2 链路包 Header 结构

每个链路包应该以一个固定大小的9字节报头开始，包括校验和，然后是一个可选的可变长度的数据负载。报头和有效负载都有它们自己的校验和。如果没有有效载荷数据，则没有有效载荷校验和。

```
0000 02 02 20 1f 00 1b 00 09 7a 0b ef 2f ff 5a 00 17 .. .....z../.Z..
0010 80 fc 00 00 14 01 01 10 00 03 e8 01 f4 0f 00 02 .....
0020 00 02 fb 9a
....
```

下面是链路包的包结构：

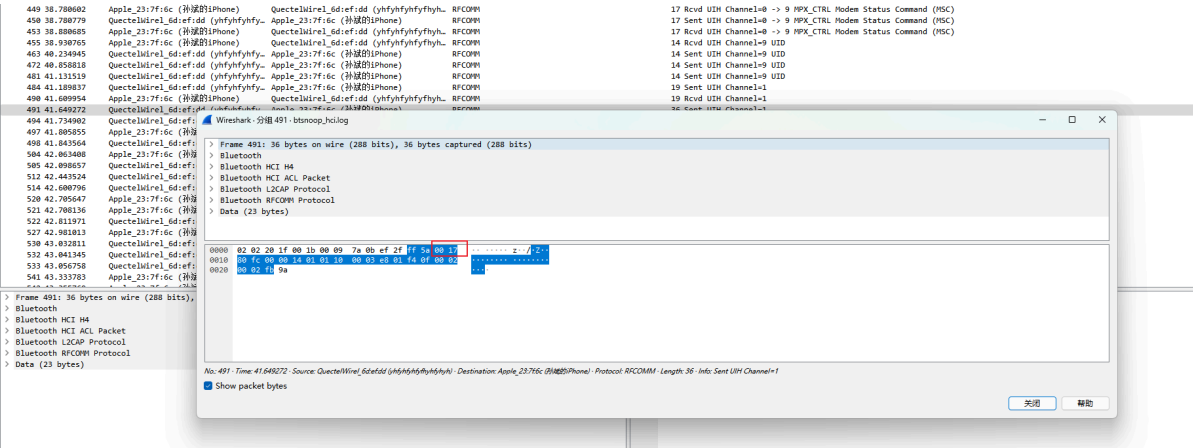
StartOfPacket MSB	StartOfPacket LSB	Length Of Packet MSB	Length Of Packet LSB	Control Byte	Packet Sequence Number	PacketSequence AckNumber	Session Identifier	Header CheckSum此部分量可选择payload数据
-------------------	-------------------	----------------------	----------------------	--------------	------------------------	--------------------------	--------------------	-----------------	-----------------------

- Start of Packet MSB
- Start of Packet LSB

包起始两个字节固定为: Start of Packet MSB = 0xFF，Start of Packet LSB = 0x5A，当存在 0xFF、0x5A 起始的字节时，后续的字节的应该被设备识别成 IAP2 报文来解析。

- Packet Length MSB
- Packet Length LSB

包 Header 的第三、四字节代表了包的长度，其中 MSB 表示大端，LSB 表示小端。2 个字节16比特，最大可以表示 0 - 65535 的长度。换句话说，IAP2 的数据包最大的长度为 65535。



从上述的蓝牙 RfComm 发送的报文中可以看到 0xFF 0x5A 之后字节是 0x00 0x17，表明包的长度是 23 字节。

- Control Byte

此字节是控制字节，标志着包的内容。控制字节存在多种不同的类型，存在有：

SYN：表示包的 Payload 内容是关于链路同步。

ACK：表示对上一个包接受到的确认/回复，并且 payload 可能会包括会话包的内容。

ESK：表示包的内容是有效的扩展。

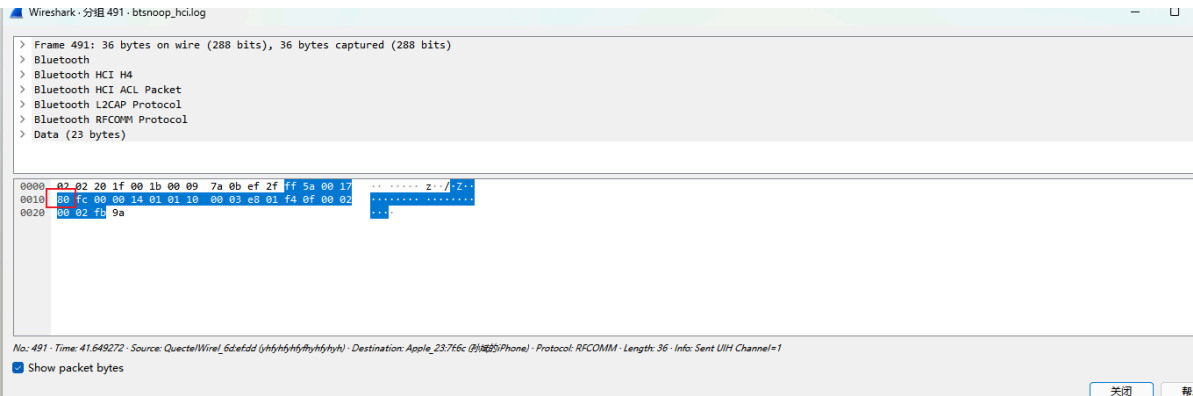
RST：表示重置此链路。

SLP：表示设备休眠。

区分控制字节的类型是通过判断对应bit位置上是否为1。

Bit	控制字节类型
7	SYN

Bit	控制字节类型
6	ACK
5	ESK
4	RST
3	SLP

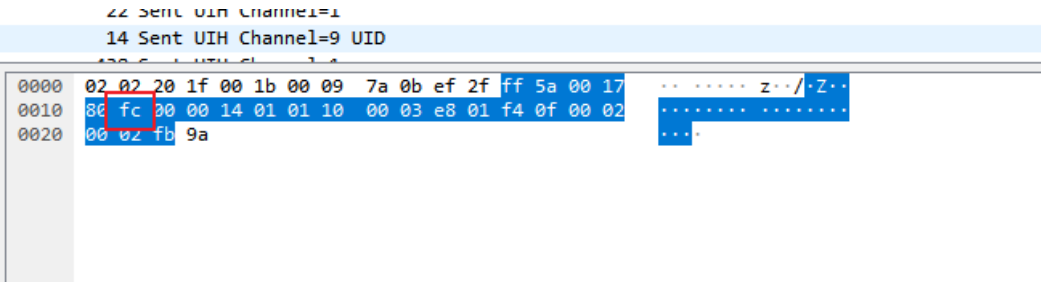


可以看到：Control Bit 的值为 0x80，转换成二进制是 1000 0000，确定第7位是为1。故可以确定此报文是 SYN，目的是用来同步的链路一些参数。

- Packet Sequence Number

控制字节之后一个字节是包序列号，每个 IAP2 包都需要包含一个包序列号，在所有正在传输中的所有其他包中唯一标识该包，一个字节表示包序列号的范围是 (0 - 255)，当序列号到达255的时候之后会从 0 开始重新计数。

一般 IAP2 链路的创建都是从附件发送控制字节为 SYN 为链路建立的开始，此时包序列号便是随机的。并且当每次包负载中含有会话的数据的时候，那么 IAP2 包的包序列号会在上一次接受到 ACK 字节位的基础增加 1，否则会保持包序列号。



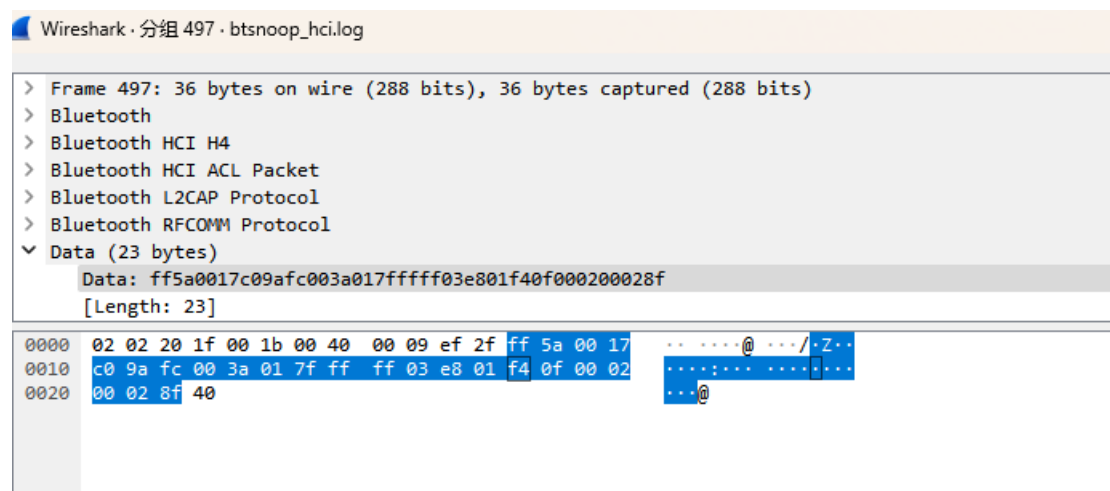
可以看到，发送 SYN 时，包序列号是随机成 fc，表示此包的包序列号是 252。关于包序列号是如何递增的，可以查看 IAP2 包解析事件。

- Packet Acknowledgement Number

Packet Sequence Number 之后的字节定义成 Ack，此字节是数据包确认位，表示对上一个报文的确认回复。

只有在设置了控制字节中的 ACK 位时，数据包确认号才有意义。例如上个 SYN 报文中，ACK 第 6 位设置为 0，那么 ACK 数据位需要设置位 00 才有意义。从 RFCOMM 中读取的数据 80 fc 00，包数据确认位设置的字节是 00。

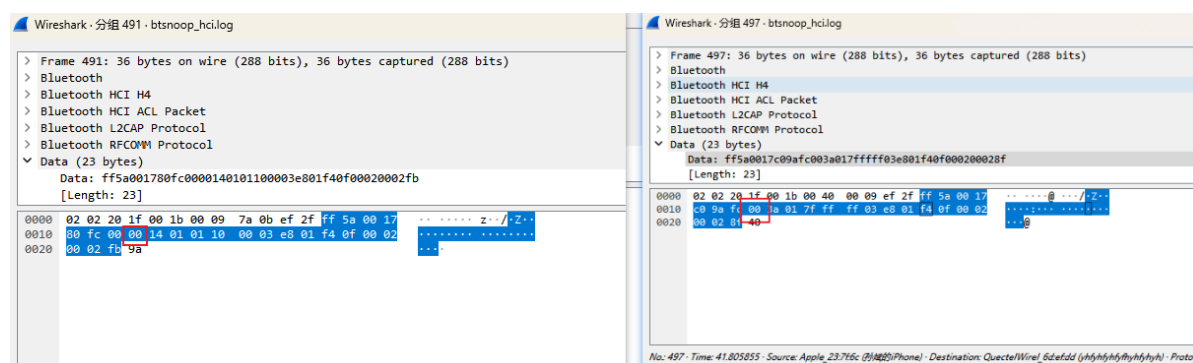
如果设置了 ACK，则表示最后接受到的一个数据包报文是此字节代表的序列号。



可以看到上面 IAP2 链路报文控制位是 C0，其中第 6、7 都是 1，表明数据包报文的类型是 SYN+ACK。表示对上面 SYN 的回复。其中 9A 是此包的序列号，SYN+ACK 的序列包和 SYN 类型包的序列号相同，一开始都是随机出来的。这里随机成 9a，ACK 字节位被设置为 fc，表示最后一个接收到数据包的序列号是 fc，表示对序列号为 fc 的回复。

- Session Identifier

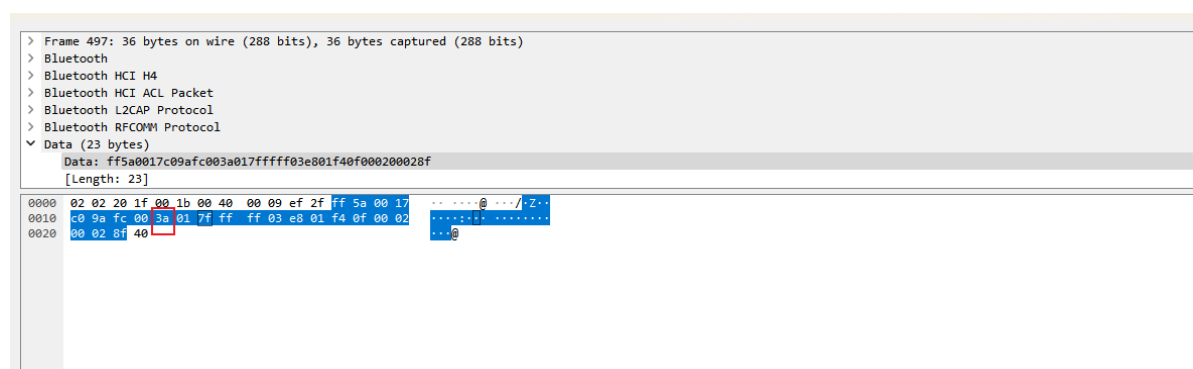
会话标识符字节，表示此 IAP2 包是否是一个控制会话包。只有在控制字节中的 ACK 位并且存在 iAP2 会话有效负载时，会话标识符才有意义。如果满足这两个条件，则会话标识符将是一个非零号，用来指定 iAP2 连接中的特定会话。否则，会话标识符应设置为 0。



SYN 和 SYN+ACK 类型数据包会话标识字节都是设置位 0x00，表示这两个 IAP2 包都是非会话包。后续 Session Identifier 为 2 可以判断为会话包，故可以通过此字段快速判断是否是会话包还是链路包。

- Header Checksum

最后一个字节是头部的校验位，用于验证 IAP2 Header 的报文是否是正确的。计算的方式是 (0x100 - 前八个字节求和之后) 取出低位字节，和此字节数据进行对比，如果对比不上，此 IAP2 报文会被丢弃。



$$0xFF + 0x5A + 0x00 + 0x17 + 0x0C + 0x9A + 0xFC + 0x00 = 0x3C6$$

$0x100 - 0x3C6 = FD3A$ 取出一个字节即为 3a，表明此报文 Header 传输时正确的。

IAP2 Session 会话详解

之前 IAP2 Header 中第 8 个字节标志了 Session 的类型，等于 0 表明此报文是控制类型的会话。

IAP2 Session 类型

Session类型	Session类型
0	控制会话
1	File Transfer session (文件传输会话)
2	External Accessory session (外部附件会话)

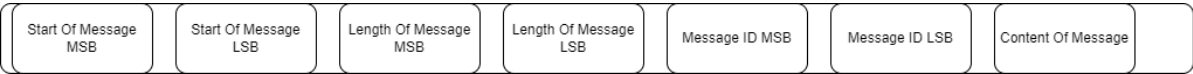
不同的会话类型处理方式是不同的，Carplay 连接的过程中主要涉及到的会话类型是 0(控制会话)和 2(外部附件会话)，下面展开也会去详细展示不同会话类型包组成的方式。通常来说，session 携带的数据在 IAP2 Header 之后的，通常是从第 10 个字节到字节流结尾。

IAP2 控制 Session 详解

IAP2 外部附件 Session 详解

Message Structure 消息格式

Session 携带数据也是按照一些固定的格式来组织的：



- Start Of Message MSB
- Start Of Message LSB

上述两个字节表示 Message 的开始，和 Header 头两个字节类似。其值是固定的，均为 0x40。

- Length Of Message MSB
- Length Of Message LSB

接下来的两个字节表示 Message 包的长度，MSB 表示字节的大端，LSB 表示字节的小端。

- Message ID MSB
- Message ID LSB

第 5、6 两个字节表示 Message Id，不同的 Message Id 是不同的会话功能的，并且携带 Parameter 同样存在类型区别。

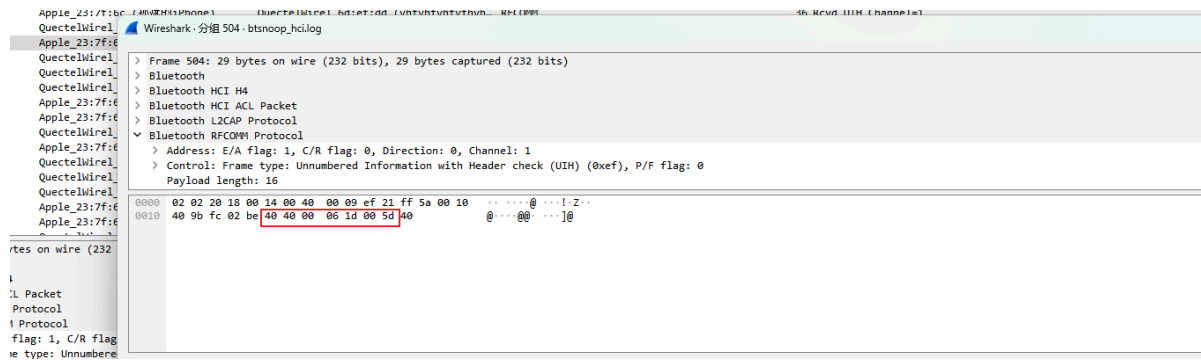
- Message Parameter

参数的类型，这个参数类型是可选择，某一些类型的 Message 不存在 Parameter，由 MessageID 定义的。具体在对应文档表中可以查询到。

Carplay 连接过程 Session

在经过握手、link 同步之后就进入到会话传输数据部分了，Carplay 建立连接过程 session 的类型都是 0x00、0x02。

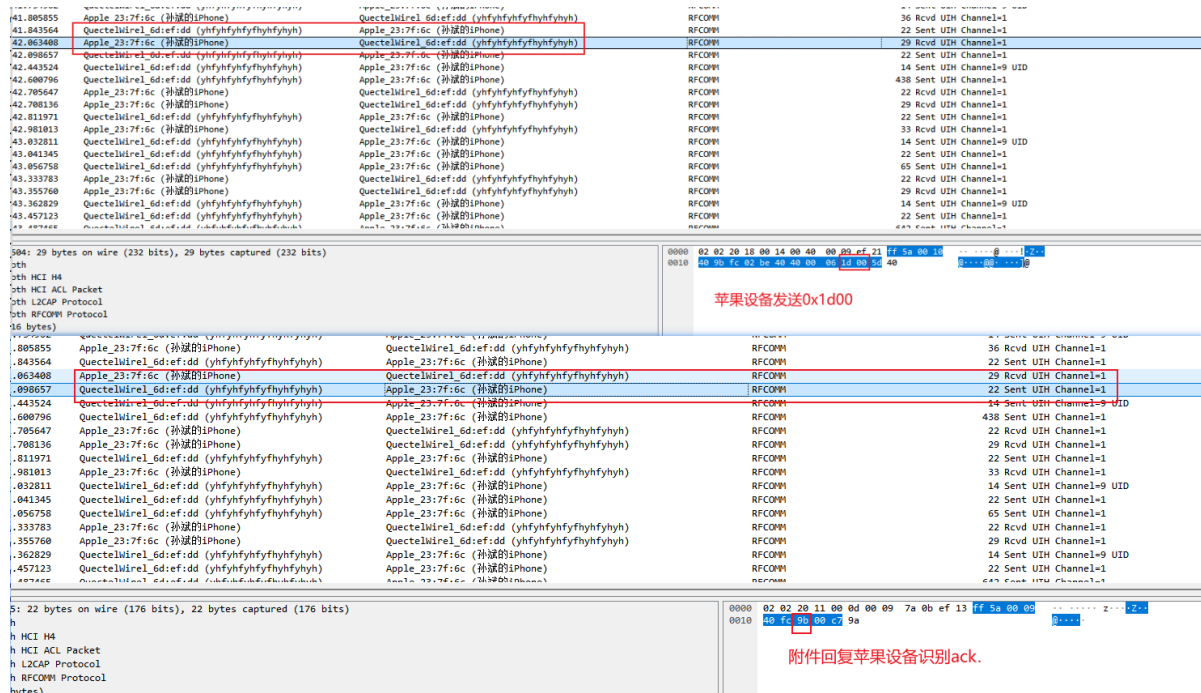
0x1d0x00 配件识别



上述的 IAP2 报文 Message 字段为 0x40 0x40 0x00 0x06 0x1d 0x00。其中 Message id 的两个字节是 0x1d 0x00，此 id 表示开始识别 StartIdentification。并且此 id 是不存在 Parameter 的，最后 0x5d 并不是 Message 的字节。

0x00 0x06 字段已经指明 Message 的长度为 6 个字节，这个长度包括从 Start Of Message MSB 到 Parameter 最后一个字节。

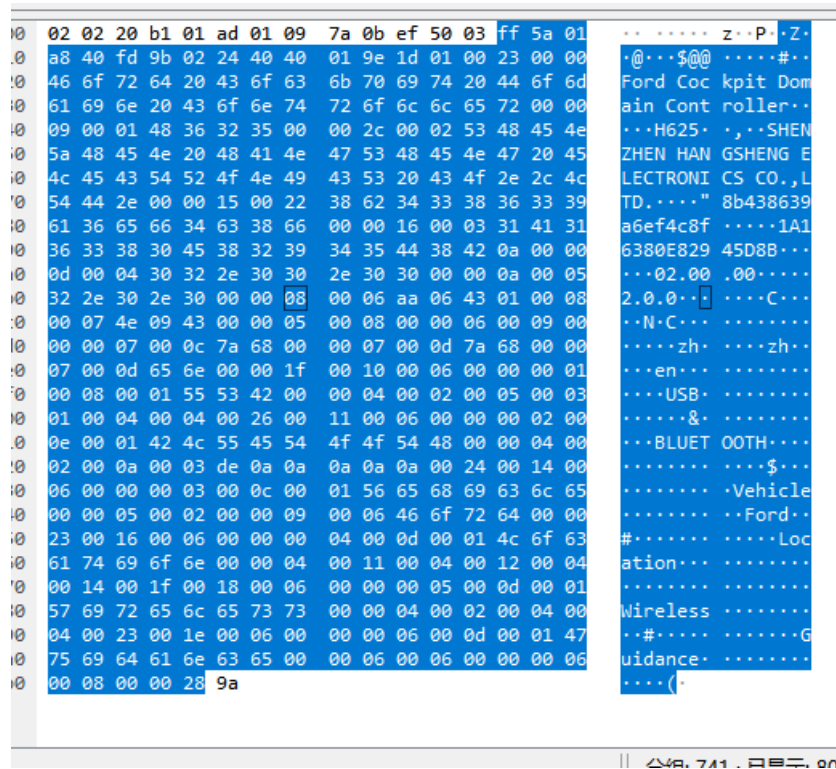
IAP2 报文中 Header 中定义了包序列号是 9b。



上面红色箭头就是表示 0x1d0x00，可以看到附件回复了对于 0x1d0x00 的回复。此 IAP2 包很简单，仅仅是对包序列号 9b 的确认 ACK。

0x1d0x01 附件识别信息

在经过 0x1d0x00 附件和设备的确定之后，就会发送 0x1d0x01 附件识别信息给 Apple 设备。下图中可以看到 Message 的长度是 0x010x9e，转换成的十进制即为 414 字节，并且包序列号是 fd。

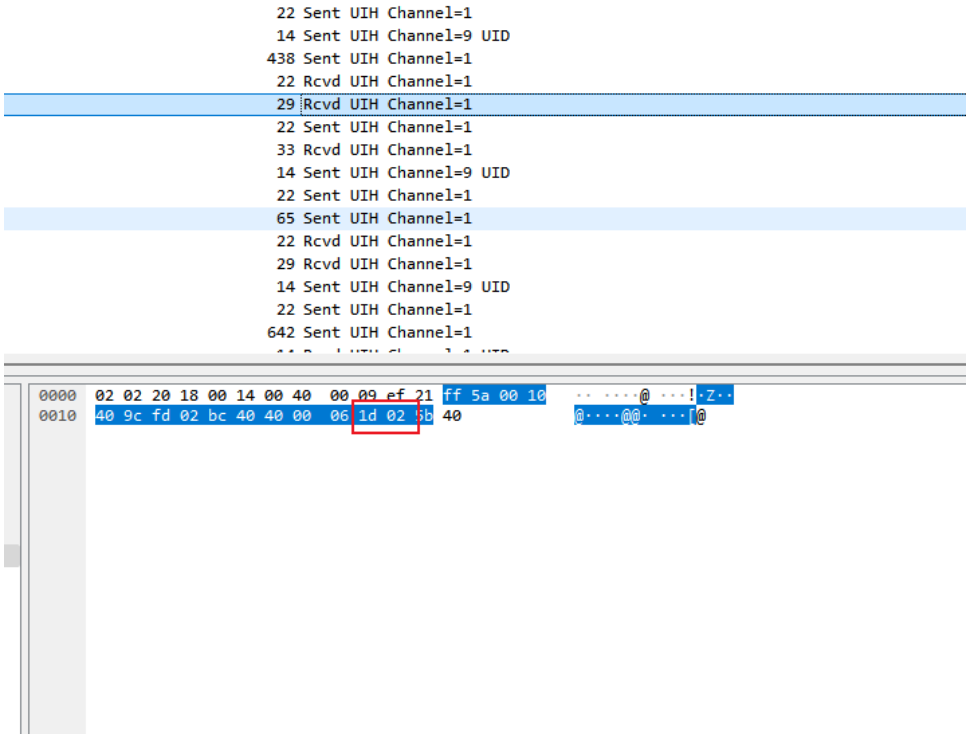


MessageId 为 0x1d0x01 的信息是携带参数的，并且携带的参数相对比较，携带的参数如下表：

Name	Id	type	Note
Name	0	utf-8	/
ModelIdentifier	1	utf-8	/
Manufacturer	2	utf-8	/
SerialNumber	3	utf-8	/
FirmwareVersion	4	utf-8	/
HardwareVersion	5	uint16	/
MessagesSentByAccessory	6	uint16	/
MessagesReceivedFromDevice	7	enum	/
PowerProvidingCapability	8	uint16	/
MaximumCurrentDrawnFromDevice	9	group	/
SupportedExternalAccessoryProtocol	10	utf-8	/
AppMatchTeamID	11	utf-8	/
CurrentLanguage	12	utf-8	/
SupportedLanguage	13	group	/
UARTTransportComponent	14	group	/
USBDeviceTransportComponent	15	group	/

Name	Id	type	Note
USBHostTransportComponent	16	group	/
BluetoothTransportComponent	17	group	/
iAP2HIDComponent	18	group	/
LocationInformationComponent	22	group	/
USBHostHIDComponent	23	group	/
BluetoothHIDComponent	29	group	/
ProductPlanUID	34	utf-8	/

如何将 `Parameter` 携带的信息解析成有效的信息，这个在 `Parameter` 的组织方式 中会详细展开。目前仅需要了解到使用 `0x1d01` 将附件标志信息发送 Apple 设备。

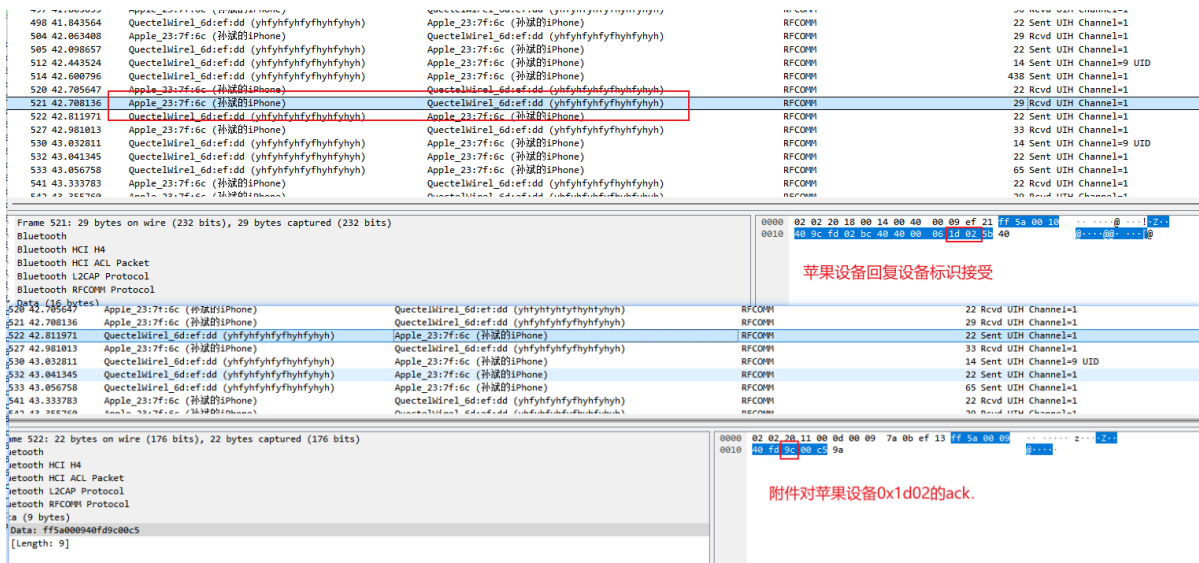


`Ack` 包对 `0x1d0x01` 回复之后就可以进入到下一流程中，表示接受到 `0x1d01` 的设备标志信息。

0x1d0x02 IdentificationAccepted 接受

经过 `0x1d01` 发送了标识信息之后，之后的 Apple 设备发送 `Accept` 或者 `Reject` 会话。其中 `0x1d0x02` 表示接受此标识，`0x1d0x03` 表示拒绝此标识。这里只讨论正确的流程也就是 `0x1d0x02`。

`0x1d0x02` 不存在 `Parameter` 的，仅仅去通知附件接受刚刚发送标识信息。

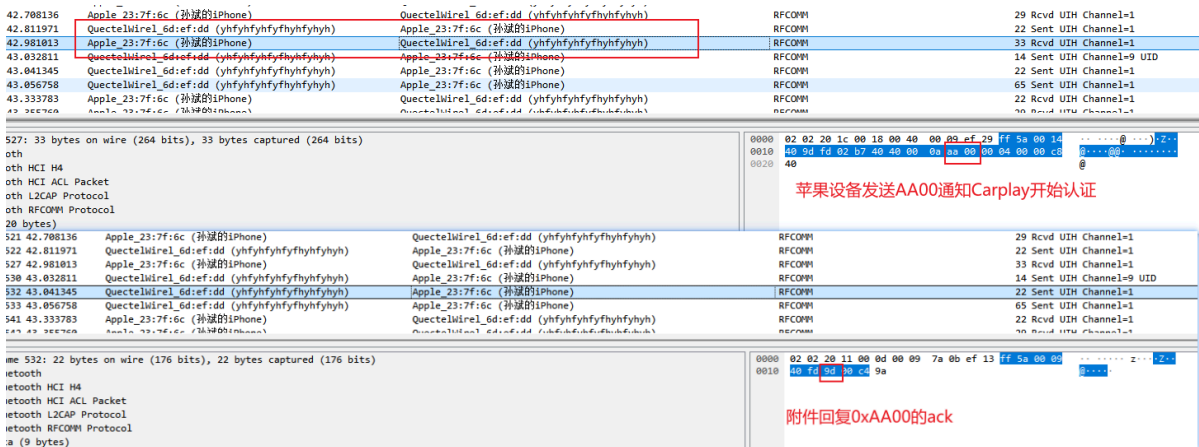


下面 IAP2 对上面 0x1d0x02 回复 ack 之后，此阶段正式结束。

经过 0x1d0x00、0x1d0x01、0x1d0x02 会话传输之后，表明设备接受到设备信息并选择接受了附件，下一步就是对附件的认证。

0xAA00 请求认证证书

此 IAP2 会话是由Apple设备向附件请求 x.509 证书的会话，此 MessageId 对应的是没有 Parameter 的，序列号为 0x9d。



0xAA06 附件认证序列号

认证过程使用 x.509 证书进行认证，需要知道 x.509 证书的序列号。附件会将 0xAA06 将 x.509 证书的序列号发送给苹果设备。Message Id AA06 存在参数，并且参数的类型是 blob，内容就是 x.509 证书序列号二进制数据。

1393	QuetcelWirel_6d:ef:dd (yhfyhyfhyfhyfhyh)	Apple_23:7f:6c (苹果的iPhone)	RFCOMM	65 Sent UIH Channel=1
6758	Apple_23:7f:6c (苹果的iPhone)	QuetcelWirel_6d:ef:dd (yhfyhyfhyfhyfhyh)	RFCOMM	22 Rcvd UIH Channel=1
5760	Apple_23:7f:6c (苹果的iPhone)	QuetcelWirel_6d:ef:dd (yhfyhyfhyfhyfhyh)	RFCOMM	22 Rcvd UIH Channel=1
2829	QuetcelWirel_6d:ef:dd (yhfyhyfhyfhyfhyh)	Apple_23:7f:6c (苹果的iPhone)	RFCOMM	14 Sent UIH Channel=9 UID
7123	QuetcelWirel_6d:ef:dd (yhfyhyfhyfhyfhyh)	Apple_23:7f:6c (苹果的iPhone)	RFCOMM	22 Sent UIH Channel=1
7400	QuetcelWirel_6d:ef:dd (yhfyhyfhyfhyfhyh)	Apple_23:7f:6c (苹果的iPhone)	RFCOMM	642 Sent UIH Channel=1

65 bytes on wire (520 bits), 65 bytes captured (520 bits)	0000 02 02 20 3c 00 38 00 09 7a 0b ef 69 ff 5a 00 34 ... <8> ... Z...
CI H4	0010 40 fe 9d 02 96 40 40 00 2a 6a 00 00 24 00 00 34 ... 8...00... \$...\$...
CI ACL Packet	0020 34 36 37 38 31 44 46 39 41 45 36 37 31 33 46 46 ... 46781DF9 AE6713FF
ZCAP Protocol	0030 36 39 43 31 44 35 30 32 33 34 38 45 36 46 45 3b ... 69C1D502 348E6FE
FCOMM Protocol	0040 9a

附件使用AA06发送认证序列号给设备

12 bytes on wire (176 bits), 22 bytes captured (176 bits)	0000 02 02 20 11 00 0d 00 40 00 09 ef 13 ff 5a 00 00 ... 8...0 ... Z...
CI H4	0010 40 9d fe 00 c3 40

苹果设备对于AA06的ack

设备回复 AA06 的 ack 之后，设备会重新发送一次 AA00 来请求 x.509 的认证证书。附件第二次接受到 AA00 请求之后，下一步会走 AA01 来传输 x.509 认证证书。

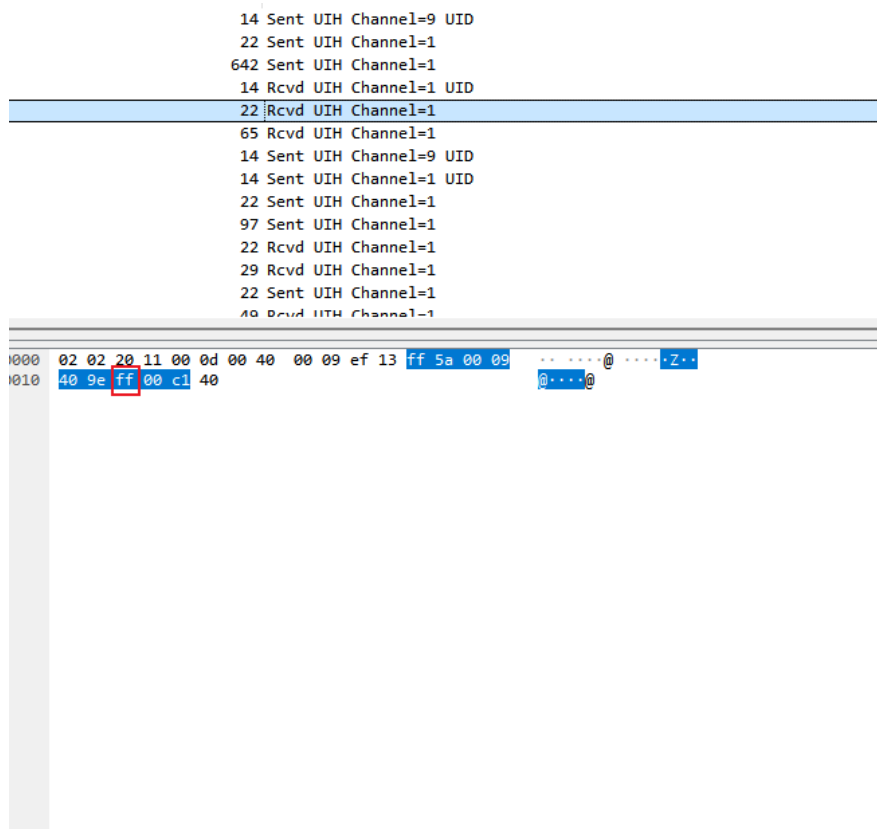
0xAA01 认证证书

IAP2 会话使用 MessageId 0xAA01 来传输附件的 x.509 的认证证书。AA01 负载就对应了附件的 x.509 证书，使用 blob 的方式来进行传输。

14 Sent UIH Channel=9 UID	22 Sent UIH Channel=1
642 Sent UIH Channel=1	14 Rcvd UIH Channel=1 UID
22 Rcvd UIH Channel=1	

00 02 02 20 7d 02 79 02 09 7a 0b ef e8 04 ff 5a 02	.. } .y . zZ..
10 74 40 ff 9e 02 52 40 40 02 6a aa 01 02 64 00 00	t@...R@...j...d..
20 30 82 02 5c 06 09 2a 86 48 86 17 0d 01 07 02 a0	0... \... * . H.....
30 82 02 4d 30 82 02 49 02 01 01 31 00 30 0b 06 09	..M0...I.. 1:0...
40 2a 86 48 86 f7 0d 01 07 01 a0 82 02 2f 30 82 02	*.H..... /0...
50 2b 30 82 01 d1 a0 03 02 01 02 02 10 44 67 81 df	+0..... :Dg..
60 9a e6 71 3f f6 9c 1d 50 23 48 e6 fe 30 0a 06 08	..q?...P #H...0...
70 2a 86 48 ce 3d 04 03 02 30 81 89 31 0b 30 09 06	*.H..... 0...1:0...
80 03 55 04 06 13 02 55 53 31 13 30 11 06 03 55 04	.U....US 1:0...U...
90 0a 13 0a 41 70 70 6c 65 20 49 6e 63 2e 31 26 30	...Apple Inc.180
a0 24 06 03 55 04 0b 13 1d 41 70 70 6c 65 20 43 65	\$..U.... Apple Ce
b0 72 74 69 66 69 63 61 74 69 6f 6e 20 41 75 74 68	rtificat ion Auth
c0 6f 72 69 74 79 31 3d 30 3b 06 03 55 04 03 13 34	ority1=0 ;..U...4
d0 41 70 70 6c 65 20 41 63 63 65 73 73 6f 72 69 65	Apple Accessorie
e0 73 20 43 65 72 74 69 66 69 63 61 74 69 6f 6e 20	s Certif ication
f0 41 75 74 68 6f 72 69 74 79 20 2d 20 30 30 30 30	Authorit y - 0000
00 30 30 30 32 30 1e 17 0d 32 34 30 32 32 37 31 30	00020... 24022710
10 30 34 34 39 5a 17 0d 34 39 31 32 33 31 32 33 35	0449Z...4 91231235
20 39 35 39 5a 30 6d 31 0b 30 09 06 03 55 04 06 13	959Z0m1. 0...U...
30 02 55 53 31 13 30 11 06 03 55 04 0a 13 0a 41 70	.US1:0... .U....Ap
40 70 6c 65 20 49 6e 63 2e 31 1a 30 18 06 03 55 04	ple Inc. 1:0...U...
50 0b 13 11 41 70 70 6c 65 20 41 63 63 65 73 73 6f	...Apple Accesso
60 72 69 65 73 31 2d 30 2b 06 03 55 04 03 14 24 49	ries1-0+ ..U...\$I
70 50 41 5f 34 34 36 37 38 31 44 46 39 41 45 36 37	PA_44678 1DF9AE67
80 31 33 46 46 36 39 43 31 44 35 30 32 33 34 38 45	13FF69C1 D502348E
90 36 46 45 30 59 30 13 06 07 2a 86 48 ce 3d 02 01	6FE0Y0... *.H=...
a0 06 08 2a 86 48 ce 3d 03 01 07 03 42 00 04 06 d5	..*.H=...B....
b0 30 7b 68 b0 39 f5 fc db ee 16 05 a7 ea 55 c5 f6	0{h.9... ..U...
c0 b6 37 24 25 c9 3f 33 71 ba f2 9a b3 6e 3c bc 8f	.7\$%?3q ...n<...
d0 f4 ce 96 00 d6 f4 8d af 57 d1 48 1d 6e be 25 50 W.H.n.%P
e0 5c cd cc 23 af c1 14 91 bd cd 0c 1b 85 60 a3 36	\..#....^..6
f0 30 34 30 32 06 09 2a 86 48 86 f7 63 64 06 24 01	0402...*. H...cd.\$.
00 01 ff 04 22 04 20 00 00 00 00 00 00 00 00 00	...".
10 00 00 00 00 00 00 00 00 00 00 00 00 00 00
20 00 00 00 00 00 10 30 0a 06 08 2a 86 48 ce 3d 040. ...*.H=...
30 03 02 03 48 00 30 45 02 20 3a 57 ef 51 0f 13 e5	...H.0E... :W.Q...
40 8c 44 5d 03 83 fc 75 9c b2 67 34 db e1 6d 4d 39	.D]...u...g4...mM9
50 e9 ce 4f c0 51 95 cd ba 56 02 21 00 b0 e5 5b a5	...O.Q... V!...[.
60 91 de 15 cb a9 14 67 62 04 ff 37 6b 8b 5f 69 32gb ..7k...i2
70 07 ac 82 4b e3 ac fb 2c c3 1e d6 c3 a1 00 31 00	...K...,1.
80 9d 9a	..

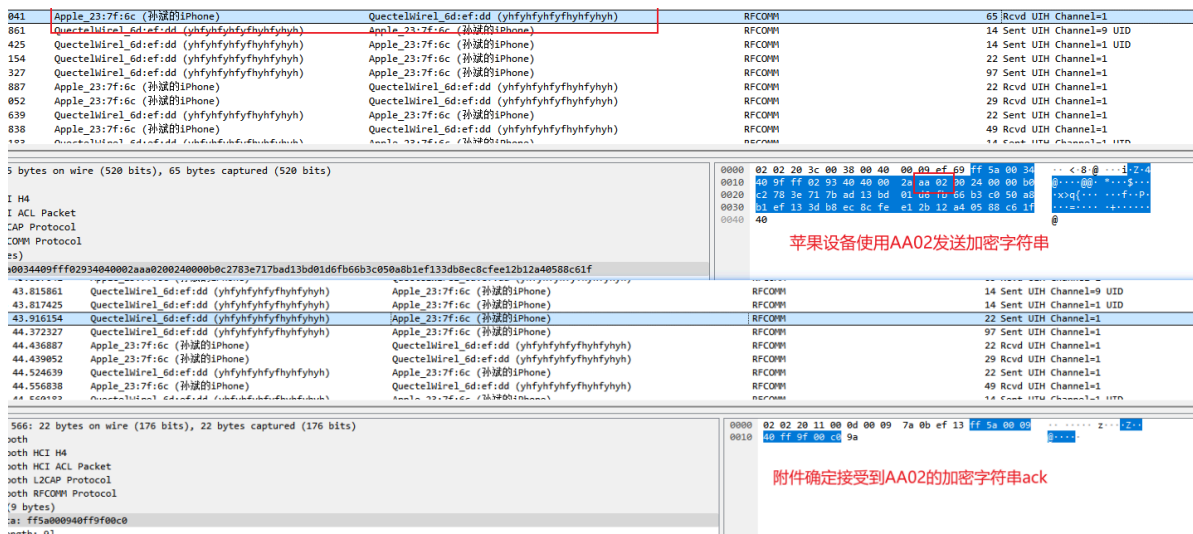
上述 IAP2 的包序列号是的是 ff，设备发送 ack 字节应该设置为 ff，抓出来 rfcomm 包是符合的并且下一个包序列号重新会包装为0。



经过 AA01、AA06 两个会话包之后，苹果设备中获取到附件的 x.509 的证书，下一步会去请求加密字符串来验证 x.509 证书。

0xAA02 请求加密字符串

苹果设备会使用 MessageId AA02 向附件传输一个需要加密的字符串。AA02 的参数类型就是字符串的二进制数据，类型同样也是 blob。



附件接受到 AA02 之后，使用 x.509 加密之后。下一步会使用 AA03 将加密之后的字符串传递给设备。

0xAA03 回复加密字符串

当接受到 AA02 的字符串之后，配件会使用证书进行加密。并将加密之后的字符串走 MessageId 0xAA03 传递给设备。

64.43.817425

Quetcelwired_6d:ef:dd (yhfyhyfhyfhyfhyfhyh)

Apple_23:7f:6c (孙诚的iPhone)

RFCOMM

14

Sent UIM Channel=1

66.43.916154

Quetcelwired_6d:ef:dd (yhfyhyfhyfhyfhyfhyh)

Apple_23:7f:6c (孙诚的iPhone)

RFCOMM

22

Sent UIM Channel=1

80.44.372327

Quetcelwired_6d:ef:dd (yhfyhyfhyfhyfhyfhyh)

Apple_23:7f:6c (孙诚的iPhone)

RFCOMM

97

Sent UIM Channel=1

87.44.436887

Apple_23:7f:6c (孙诚的iPhone)

Quetcelwired_6d:ef:dd (yhfyhyfhyfhyfhyfhyh)

RFCOMM

22

Rcvd UIM Channel=1

88.44.439852

Apple_23:7f:6c (孙诚的iPhone)

Quetcelwired_6d:ef:dd (yhfyhyfhyfhyfhyfhyh)

RFCOMM

29

Rcvd UIM Channel=1

90.44.524639

Quetcelwired_6d:ef:dd (yhfyhyfhyfhyfhyfhyh)

Apple_23:7f:6c (孙诚的iPhone)

RFCOMM

22

Sent UIM Channel=1

95.44.556838

Apple_23:7f:6c (孙诚的iPhone)

Quetcelwired_6d:ef:dd (yhfyhyfhyfhyfhyfhyh)

RFCOMM

49

Rcvd UIM Channel=1

97.44.560183

Quetcelwired_6d:ef:dd (yhfyhyfhyfhyfhyfhyh)

Apple_23:7f:6c (孙诚的iPhone)

RFCOMM

14

Sent UIM Channel=1

99.44.609722

Quetcelwired_6d:ef:dd (yhfyhyfhyfhyfhyfhyh)

Apple_23:7f:6c (孙诚的iPhone)

RFCOMM

22

Sent UIM Channel=1

100.44.907130

Apple_23:7f:6c (孙诚的iPhone)

Quetcelwired_6d:ef:dd (yhfyhyfhyfhyfhyfhyh)

RFCOMM

98

Rcvd UIM Channel=1

ne 580:

97 bytes on wire (776 bits), 97 bytes captured (776 bits) on stooth

stooth HCI H4

stooth HCI ACL Packet

stooth L2CAP Protocol

stooth RFCOMM Protocol

s (84 bytes)

[hex]: ff80a5440009f027240400aaab30044000ee8c92515b9d0e673a5cdcd1d55c2dd3c253cf7645fec5b84f3ec281c37992f277986bea54b77fb5f8

length: 84

0000

02 02 20 5c 00 58 00 09 7a 0b c0 9f

ff 5a 00 54

0010

40 9f 02 72 40 40 09 4a aa 03 80 44 00 0e

8c 09 25 15 b9 0d 54 72 85 c9 cc 1d 53 52 d6

c2 53 cf c7 64 5f ec 5b 84 f8 3e c2 01 c3 79 92

f2 77 98 6b ea 54 b7 77 bf 58 63 03 d6 b1 bf 9e

0050 05 97 ec ab 64 be 4c 2e 85 08 cc 27 66 65 0b 4c

0060 7a

附件将使用证书加密之后的字符串发送苹果设备 (0xA03)

Quetcelwired_6d:ef:dd (yhfyhyfhyfhyfhyfhyh)

Apple_23:7f:6c (孙诚的iPhone)

RFCOMM

22

Sent UIM Channel=1

Quetcelwired_6d:ef:dd (yhfyhyfhyfhyfhyfhyh)

Apple_23:7f:6c (孙诚的iPhone)

RFCOMM

97

Sent UIM Channel=1

Apple_23:7f:6c (孙诚的iPhone)

Quetcelwired_6d:ef:dd (yhfyhyfhyfhyfhyfhyh)

RFCOMM

22

Rcvd UIM Channel=1

Apple_23:7f:6c (孙诚的iPhone)

Quetcelwired_6d:ef:dd (yhfyhyfhyfhyfhyfhyh)

RFCOMM

29

Rcvd UIM Channel=1

Quetcelwired_6d:ef:dd (yhfyhyfhyfhyfhyfhyh)

Apple_23:7f:6c (孙诚的iPhone)

RFCOMM

22

Sent UIM Channel=1

Apple_23:7f:6c (孙诚的iPhone)

Quetcelwired_6d:ef:dd (yhfyhyfhyfhyfhyfhyh)

RFCOMM

49

Rcvd UIM Channel=1

Quetcelwired_6d:ef:dd (yhfyhyfhyfhyfhyfhyh)

Apple_23:7f:6c (孙诚的iPhone)

RFCOMM

14

Sent UIM Channel=1

Quetcelwired_6d:ef:dd (yhfyhyfhyfhyfhyfhyh)

Apple_23:7f:6c (孙诚的iPhone)

RFCOMM

22

Sent UIM Channel=1

Apple_23:7f:6c (孙诚的iPhone)

Quetcelwired_6d:ef:dd (yhfyhyfhyfhyfhyfhyh)

RFCOMM

98

Rcvd UIM Channel=1

res on wire (176 bits), 22 bytes captured (176 bits) on stooth

.L Packet

Protocol

f Protocol

0000

02 02 20 11 00 0d 00 40 00 09 ef 13 ff 5a 00 54

0010

40 9f 02 72 40 40 09 4a aa 03 80 44 00 0e

苹果设备回复0xA03ack

0940f00000f

这里可以看到，此 IAP2 包的包序列号从 ff 最大值重新被包装成 00 了。

AA03 将附件加密之后的字符串发送给设备之后，设备会进行解密之后和发送的数据进行对比。如果相同的话，设备会发送 AA05 表示认证成功，如果不相同会发送 AA04 表示认证不成功。

0xAA05 认证成功

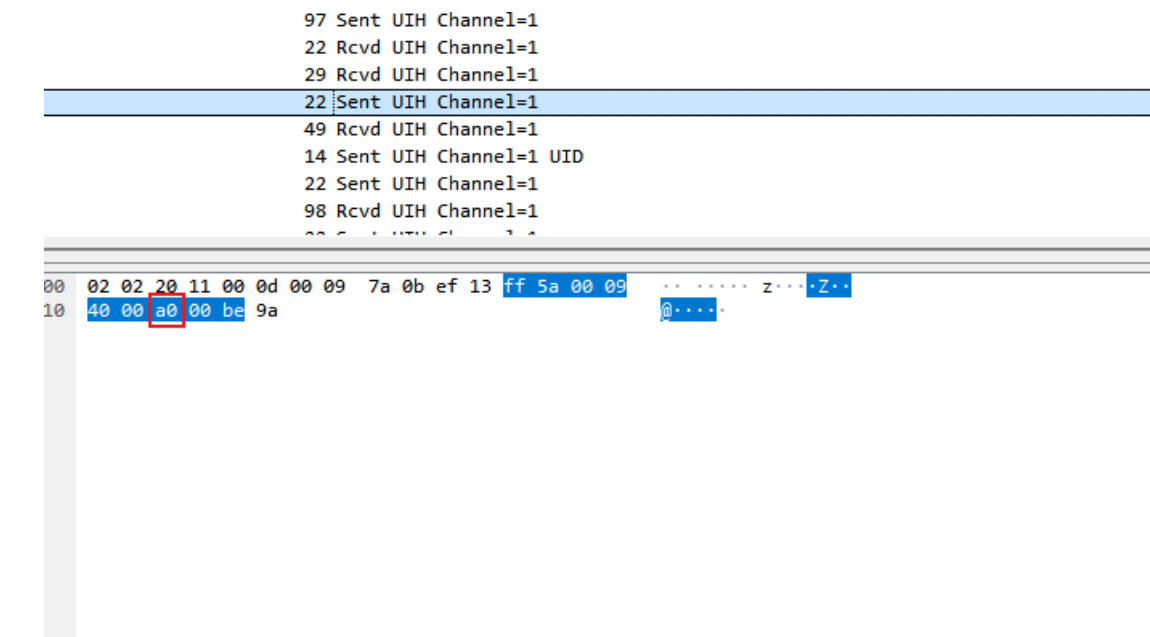
此 MessageId 为 0xAA05 的时候表示认证成功，0xAA05 是不携带 Parameter 的仅仅去通知附件认证成功。

```

22 Sent UIH Channel=1
97 Sent UIH Channel=1
22 Rcvd UIH Channel=1
29 Rcvd UIH Channel=1
22 Sent UIH Channel=1
49 Rcvd UIH Channel=1
14 Sent UIH Channel=1 UID
22 Sent UIH Channel=1
98 Rcvd UIH Channel=1
00 Sent UIH Channel=1
0000 02 02 20 18 00 14 00 40 00 09 ef 21 ff 5a 00 10 .. ...@ ...! Z..
0010 40 a0 00 02 b5 40 40 00 06 aa 05 cb 40 @...@...@

```

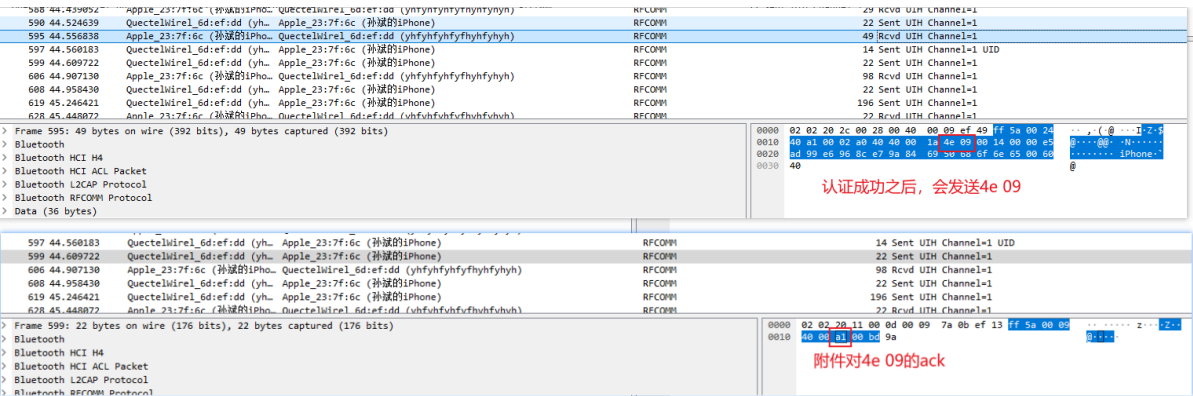
当前的包序列号是 0xa0，ack 期待的字段即为 0xa0。



到此为止，整个认证会话的流程结束了。

0x4E09 更新设备信息

认证成功之后，苹果设备会发送 MessageId 为 0x4E0x09 来更新设备名称，下面的 Rfcomm 报文将设备名称更新为 iPhone。



下个阶段会进入到 Carplay 特有的会话 session 中，主要是传输热点的一些配置信息。

0x4300 carplay Availabilty 可用性

MessageId 等于 0x4300 来指示 Carplay 可用性。在接受 0x4e09 更新设备名称之后，Apple 设备会使用 0x4300 来通知附件 Carplay 可用。0x4300会携带参数，携带的参数为如下表：

Name	ID	Type	Note
WiredAttributes	0	group	通过USB显示CarPlay的可用性
WirelessAttributes	1	group	表示在无线网络上提供的CarPlay的可用性

wiredAttributes 会携带两个参数

Name	ID	Type	Note
Available	0	bool	有线 Carplay 的可用性

Name	ID	Type	Note
USBTransportIdentifier	1	utf-8	USB传输标识符

wirelessAttributes 也会携带两个参数

Name	ID	Type	Note
Available	0	bool	无线 Carplay 的可用性
BluetoothTransportIdentifier	1	utf-8	蓝牙传输标识符

下面是 0x4300 的原始 IAP2 的完整报文。

0000	02 02 20 5d 00 59 00 40 00 09 ef ab ff 5a 00 55	..].Y.@.....Z.U
0010	40 a2 00 02 6e 40 40 00 4b 43 00 00 26 00 00 00	@...n@@.KC..&...
0020	05 00 00 01 00 1d 00 01 30 30 30 30 38 31 30 3100008101
0030	30 30 31 35 36 38 32 39 31 45 34 34 30 30 31 45	001568291E44001E
0040	00 00 1f 00 01 00 05 00 00 01 00 16 00 01 35 635c
0050	3a 38 37 3a 33 30 3a 32 33 3a 37 66 3a 36 63 00	:87:30:23:7f:6c.
0060	67 40	g@

0x430x00 之后的两个参数表示 Group0 的长度，0x0026 是 wiredAttributes 字段的长度，换算成十进制即为38。

	00 26 00 00 00	@...n@@.KC..&...
0020	05 00 00 01 00 1d 00 01 30 30 30 30 38 31 30 3100008101
0030	30 30 31 35 36 38 32 39 31 45 34 34 30 30 31 45	001568291E44001E
0040	00	

接下来的两个字节就是 0x0000 表示 groupid，这里代表是 group Id 为 0。下面就是 wiredAttributes 组了，0x0005 表示 wiredAttributes 第一个参数的长度为 5 个字节，字节片段如下：

00 05 00 00 01

其中 0x0000 表示 group 第一个参数即 Available，其值为：01；表示有线 Carplay 是可用的，接下来的两个字节表示第二个参数的长度，001d 表示 USBTransportIdentifier 的长度，有 29 个字节。后续字节 0x0001 表示 USBTransportIdentifier 的 id,之后的25个字节即为 USBTransportIdentifier 的内容，转换成Ascii对应的值就是有线连接的 USB 传输标识符。

	00 1d 00 01 30 30 30 30 38 31 30 3100008101
0030	30 30 31 35 36 38 32 39 31 45 34 34 30 30 31 45	001568291E44001E
0040	00	

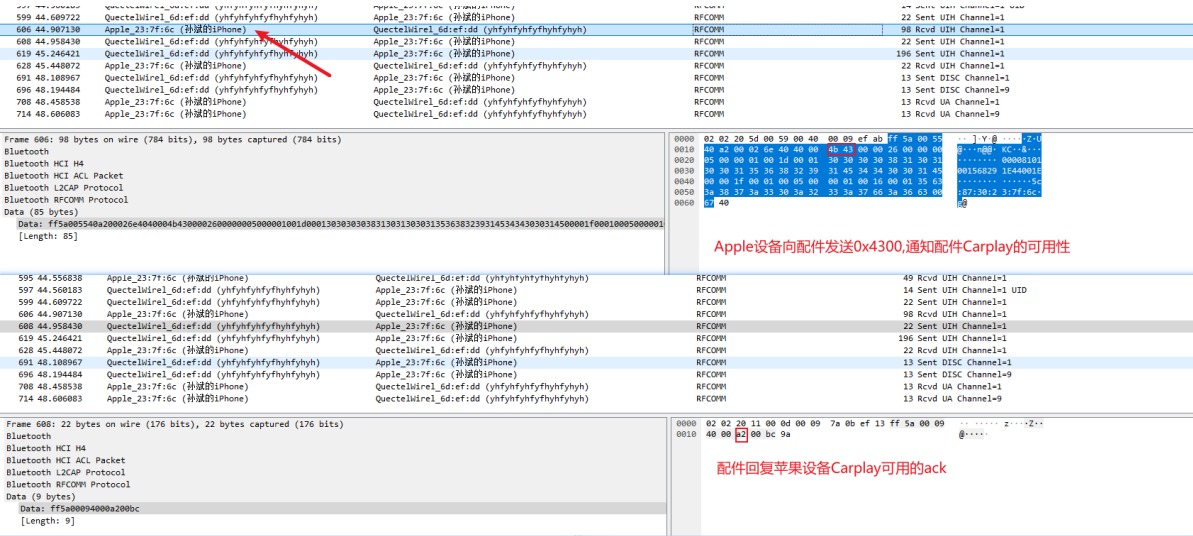
紧接着就是group1的长度，0x001f就是 wirelessAttributes 字段的长度，换算成十进制即为31。

	00 1f 00 01 00 05 00 00 01 00 16 00 01 35 635c
0050	3a 38 37 3a 33 30 3a 32 33 3a 37 66 3a 36 63 00	:87:30:23:7f:6c.
0060	67 40	g@

接下来的两个字节就是 0x0001 表示 groupid，这里代表是 group Id 为 1。下面就是 WirelessAttributes 组了，0x0005 表示 wiredAttributes 第一个参数的长度为 5 个字节，字节片段如下：

```
00 05 00 00 01
```

其中 0x0001 表示 Available 的 Id，其值为：01；表示无线 Carplay 是可用的，接下来的两个字节表示第二个参数的长度，0016 表示 BluetoothTransportIdentifier 的长度，22 个字节。后续字节 0x0001 表示 BluetoothTransportIdentifier 的 id，之后的 20 个字节即为 BluetoothTransportIdentifier 的内容，转换成 Ascii 对应的值就是无线连接的蓝牙传输标识符。



附件接受到 Apple 设备 Carplay 可用之后，会回复 0x4301 来将热点信息给的苹果设备。

0x4301 开始 Carplay 会话

附件接受到 4300Carplay 可用性之后会发送 4301 来开启 Carplay 会话。0x4301 会携带参数，其参数列表如下：

Name	ID	Type	Note
WiredAttributes	0	group	如果 4300 支持有线 Carplay 的时候，那么此属性必须包含
WirelessAttributes	1	group	如果 4300 支持无线 Carplay 的时候，那么此属性必须包含
Port	2	uint32	
DeviceIdentifier	3	utf-8	附件的全球唯一ID
PublicKey	4	utf-8	配对标识字符串
SourceVersion	5	utf-8	苹果CarPlay通信插件的源版本

wiredAttributes group 参数如下：

Name	ID	Type	Note
IPAddress	0	utf8	有线连接的IPv6地址

wirelessAttributes group 参数如下:

Name	ID	Type	Note
WiFiSSID	0	utf8	WIFI的SSID
Passphrase	1	utf8	WIFI的密码
Channel	2	uint8	WIFI连接信道
IPAddress	3	utf-8	WIFI的IP地址
SecurityType	4	uint8	WIFI的安全类型, 枚举对应的值

```
0000                                     ff 5a 00  .. ..z..1..Z.
0010  b6 40 01 a2 02 0c 40 40 00 ac 43 01 00 53 00 01  .@....@@...C..S..
0020  00 13 00 00 41 6e 64 72 6f 69 64 41 50 5f 39 38  ....AndroidAP_98
0030  30 33 00 00 14 00 01 79 78 7a 70 63 61 6e 77 79  03.....yxzpcanwy
0040  64 67 69 33 76 63 00 00 05 00 02 95 00 1e 00 03  dgi3vc.....
0050  66 65 38 30 3a 3a 34 36 31 61 3a 38 34 66 66 3a  fe80::461a:84ff:
0060  66 65 36 64 3a 66 66 38 39 00 00 05 00 04 03 00  fe6d:ff89.....
0070  08 00 02 00 00 1b 58 00 0c 00 05 34 37 35 2e 38  .....X....475.8
0080  2e 31 00 00 16 00 03 38 44 3a 32 30 3a 44 33 3a  .1.....8D:20:D3:
0090  41 39 3a 42 42 3a 43 31 00 00 29 00 04 61 39 39  A9:BB:C1..)..a99
00a0  34 34 61 30 34 2d 66 61 31 39 2d 34 63 34 33 2d  44a04-fa19-4c43-
00b0  62 65 33 36 2d 64 65 63 34 65 39 64 30 39 37 38  be36-dec4e9d0978
00c0  66 00 89 9a                                     f...
```

对于 4301 携带的参数首先可以确定不同的参数 Id 对应字节流如下, 由于是无线 CarpaLy 连接的。故第一个参数 Id == 0 是缺少的, 并且可以发现参数的排列也是不是按照顺序来的, 下面的字节流是 01、02、05、03、04 的顺序, 依次是 wirelessAttributes、Port、SourceVersion、SourceVersion、SecurityType。

下面是 wirelessAttributees 的字节流:

```
                                00 53 00 01  .@....@@...C..S..
0020  00 13 00 00 41 6e 64 72 6f 69 64 41 50 5f 39 38  ....AndroidAP_98
0030  30 33 00 00 14 00 01 79 78 7a 70 63 61 6e 77 79  03.....yxzpcanwy
0040  64 67 69 33 76 63 00 00 05 00 02 95 00 1e 00 03  dgi3vc.....
0050  66 65 38 30 3a 3a 34 36 31 61 3a 38 34 66 66 3a  fe80::461a:84ff:
0060  66 65 36 64 3a 66 66 38 39 00 00 05 00 04 03

//00 13 00 00 41 6e 64 72 6f 69 64 41 50 5f 39 38 30 33 00 // 对应
wirelessAttributes Id == 0即WIFI SSID

//00 14 00 01 79 78 7a 70 63 61 6e 77 79 64 67 69 33 76 63 00 //对应
wirelessAttributes Id == 1即WIFI Passphrase

//00 05 00 02 95 //对应wirelessAttributes Id == 2即WIFI Channel

00 1e 00 03 66 65 38 30 3a 3a 34 36 31 61 3a 38 34 66 66 3a 66 65 36 64 3a 66 66
38 39 00 // 对应wirelessAttributes ID == 3即WIFI IPAddress
```

00 05 00 04 03//对应wirelessAttributes Id == 4即WIFI PSK加密类型。目前文档中写明的支持方式是存在两种，一种是03(WPA3个人过度模式)，一种是04(WPA3个人模式)

下面是 Port 的字节流：

00 08 00 02 00 00 1b 58
//Port端口的值为1b58换算成十进制即为：7000

下面是 SourceVersion 的字节流：

00 0c 00 05 34 37 35 2e 38 2e 31 00
转换成字符串即：475.8.1

下面是 DeviceIdentifier 的字节流：

00 16 00 03 38 44 3a 32 30 3a 44 33 3a 41 39 3a 42 42 3a 43 31 00
转换成字符串即为：8D:20:D3:A9:BB:C1

下面是 PublicKey 的字节流：

00 29 00 04 61 39 39 A9:BB:C1...)..a99
00a0 34 34 61 30 34 2d 66 61 31 39 2d 34 63 34 33 2d 44a04-fa19-4c43-
00b0 62 65 33 36 2d 64 65 63 34 65 39 64 30 39 37 38 be36-dec4e9d0978
00c0 66 00
转换成字符串就是：a9944a04-fa19-4c43-be36-dec4e9d0978f

597 44.568183	Quetcelkirel_6d:ef:dd (yhfyhfyhfyhfyhfyh)	Apple_23:7f:6c (孙斌的iPhone)	RFCOMM	14 Sent UIH Channel=1 UID
599 44.609722	Quetcelkirel_6d:ef:dd (yhfyhfyhfyhfyhfyh)	Apple_23:7f:6c (孙斌的iPhone)	RFCOMM	22 Sent UIH Channel=1
606 44.907138	Apple_23:7f:6c (孙斌的iPhone)	Quetcelkirel_6d:ef:dd (yhfyhfyhfyhfyhfyh)	RFCOMM	98 Rcvd UIH Channel=1
608 44.958438	Quetcelkirel_6d:ef:dd (yhfyhfyhfyhfyhfyh)	Apple_23:7f:6c (孙斌的iPhone)	RFCOMM	22 Sent UIH Channel=1
619 45.246421	Quetcelkirel_6d:ef:dd (yhfyhfyhfyhfyhfyh)	Apple_23:7f:6c (孙斌的iPhone)	RFCOMM	196 Sent UIH Channel=1
628 45.448072	Apple_23:7f:6c (孙斌的iPhone)	Quetcelkirel_6d:ef:dd (yhfyhfyhfyhfyhfyh)	RFCOMM	22 Rcvd UIH Channel=1
691 48.108967	Quetcelkirel_6d:ef:dd (yhfyhfyhfyhfyhfyh)	Apple_23:7f:6c (孙斌的iPhone)	RFCOMM	13 Sent DISC Channel=1
696 48.194484	Quetcelkirel_6d:ef:dd (yhfyhfyhfyhfyhfyh)	Apple_23:7f:6c (孙斌的iPhone)	RFCOMM	13 Sent DISC Channel=9
708 48.458538	Apple_23:7f:6c (孙斌的iPhone)	Quetcelkirel_6d:ef:dd (yhfyhfyhfyhfyhfyh)	RFCOMM	13 Rcvd UA Channel=1
714 48.606083	Apple_23:7f:6c (孙斌的iPhone)	Quetcelkirel_6d:ef:dd (yhfyhfyhfyhfyhfyh)	RFCOMM	13 Rcvd UA Channel=9

Frame 619: 196 bytes on wire (1568 bits), 196 bytes captured (1568 bits)
Bluetooth
Bluetooth HCI H4
Bluetooth HCI ACL Packet
Bluetooth L2CAP Protocol
Bluetooth RFCOMM Protocol
Data (182 bytes)
Data [..]: ffsa00b64801a2020c404000ac43010053000100130000416e54726f696441505f3938303000014000179787a7863616e7779646769337663000005
[Length: 182]

0000 02 02 20 bf 00 bb 00 09 7a 0b ef 6c 01 ff 5a 00z-1-2-
0010 b6 40 01 a2 02 0c 40 40 00 ac 43 01 00 53 00 01@...C...5-
0020 80 13 00 00 41 6e 64 72 6f 69 64 4e 50 57 39 38Andr oid00_36
0030 80 33 00 00 14 00 01 79 76 7a 70 63 6a 6e 77 76xy xpcamoy
0040 64 67 69 33 76 63 00 00 05 00 02 95 00 1e 00 03g13vc.....
0050 66 65 38 30 3a 3a 34 36 31 61 3a 38 34 66 66 3afe00:46 1a:84ff:
0060 65 65 36 64 3a 66 6c 38 39 00 00 05 00 0a 03 00fedd:fffb 9.....
0070 80 00 02 00 00 1b 58 00 0c 00 05 34 37 35 2e 38X...475.8
0080 2e 31 00 00 16 00 03 38 44 3a 32 30 3a 44 33 3a1.....8 D:20:D3:
0090 41 39 3a 42 42 3a 43 31 00 00 29 00 04 61 39 39A9:BB(C1...)..a99
00a0 34 34 61 30 34 2d 66 61 31 39 2d 34 63 34 33 2d44a04-fa 19-4c43-
00b0 62 65 33 36 2d 64 65 63 34 65 39 64 30 39 37 38be36-dec 4e9d0978
00c0 66 00 00 00f

使用0x4301发送开启CarplaySession

Frame 628: 22 bytes on wire (176 bits), 22 bytes captured (176 bits)
Bluetooth
Bluetooth HCI H4
Bluetooth HCI ACL Packet
Bluetooth L2CAP Protocol
Bluetooth RFCOMM Protocol
Data (9 bytes)
Data [..]: ffsa00b0940a20100bb
[Length: 9]

0000 02 02 20 11 00 0d 00 40 00 ef 13 ff 5a 00 00@
0010 40 a2 01 00 b7 40@.....

对Carplay Start Session开启会话的回复。

整个Carplay交互流程

