**iAuto for UAES**

**MPU solution**

**<DoIP用户手册>**

**2019/10/17**

**History**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 版本号 | 变更内容 | 变更者 | | 日期 | 审核人 | |
| 1.0 | **Initial Version** | | **An Xuecheng** | **2019/10/15** | |  |
| 1.1 | **追加异常系说明** | | **An Xuecheng** | **2019/10/17** | |  |

目录

[1 DoIP工作原理 6](#_Toc22236169)

[2 DoIP Application实现机制 8](#_Toc22236170)

[2.1 UDS Client 8](#_Toc22236171)

[2.2 UDSServer 9](#_Toc22236172)

[2.3 UDS Client/ UDS Server 9](#_Toc22236173)

[3 DoIP SDK简介 10](#_Toc22236174)

[3.1 bin目录 10](#_Toc22236175)

[3.2 config目录 10](#_Toc22236176)

[3.3 include目录 11](#_Toc22236177)

[3.4 lib目录 11](#_Toc22236178)

[3.5 demo目录 11](#_Toc22236179)

[4 DoIP开发说明 12](#_Toc22236180)

[4.1 DoIP协议栈服务端实现说明 12](#_Toc22236181)

[4.1.1 初始化协议栈 12](#_Toc22236182)

[4.1.2 ECU情报传入 12](#_Toc22236183)

[4.1.3 使能线激活及非激活 12](#_Toc22236184)

[4.1.4 有效IP分配 13](#_Toc22236185)

[4.1.5 ECU Ready状态 13](#_Toc22236186)

[4.2 DoIP 应用层UDS Client端实现说明 13](#_Toc22236187)

[4.2.1 初始化与销毁 13](#_Toc22236188)

[4.2.2 DoIP请求发送 14](#_Toc22236189)

[4.3 DoIP 应用层UDS Server端实现说明 14](#_Toc22236190)

[4.3.1 初始化与销毁 15](#_Toc22236191)

[4.3.2 DoIP请求回应 15](#_Toc22236192)

[5 DoIP安装部署 16](#_Toc22236193)

[5.1 角色说明 16](#_Toc22236194)

[5.2 程序部署说明 16](#_Toc22236195)

[5.3 程序运行 17](#_Toc22236196)

[5.3.1 网络配置 17](#_Toc22236197)

[5.3.2 启动uds server（以太网ECU） 17](#_Toc22236198)

[5.3.3 启动uds client（MPU） 18](#_Toc22236199)

[5.4 程序日志分析 19](#_Toc22236200)

[5.4.1 MPU端的udsclient日志（./bin/udsclient） 19](#_Toc22236201)

[5.4.2 ECU端的udsserver日志（./bin/udsserver） 20](#_Toc22236202)

[5.4.3 MPU端的doipdaemon （./bin/doipdaemon） 21](#_Toc22236203)

[5.4.4 ECU端的doipdaemon （./bin/doipdaemon） 25](#_Toc22236204)

[5.5 注意事项 27](#_Toc22236205)

[6 DoIP抓包与报文分析 28](#_Toc22236206)

[6.1 通过 wireshark 工具进行抓包 28](#_Toc22236207)

[6.2 DoIP报文分析 28](#_Toc22236208)

[6.2.1 DoIP车辆公告报文分析 29](#_Toc22236209)

[6.2.2 DoIP车辆发现报文分析 30](#_Toc22236210)

[6.2.3 DoIPPowermode报文分析 31](#_Toc22236211)

[6.2.4 DoIPEntitystatus报文分析 33](#_Toc22236212)

[6.2.5 DoIProuting activation报文分析 35](#_Toc22236213)

[6.2.6 DoIPDiagnostic message报文分析 37](#_Toc22236214)

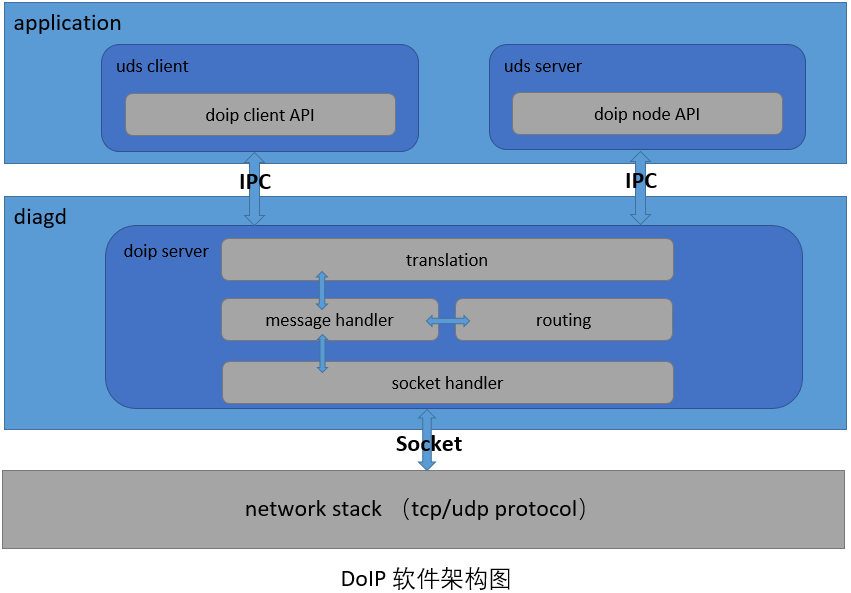
[7 异常系处理 41](#_Toc22236215)

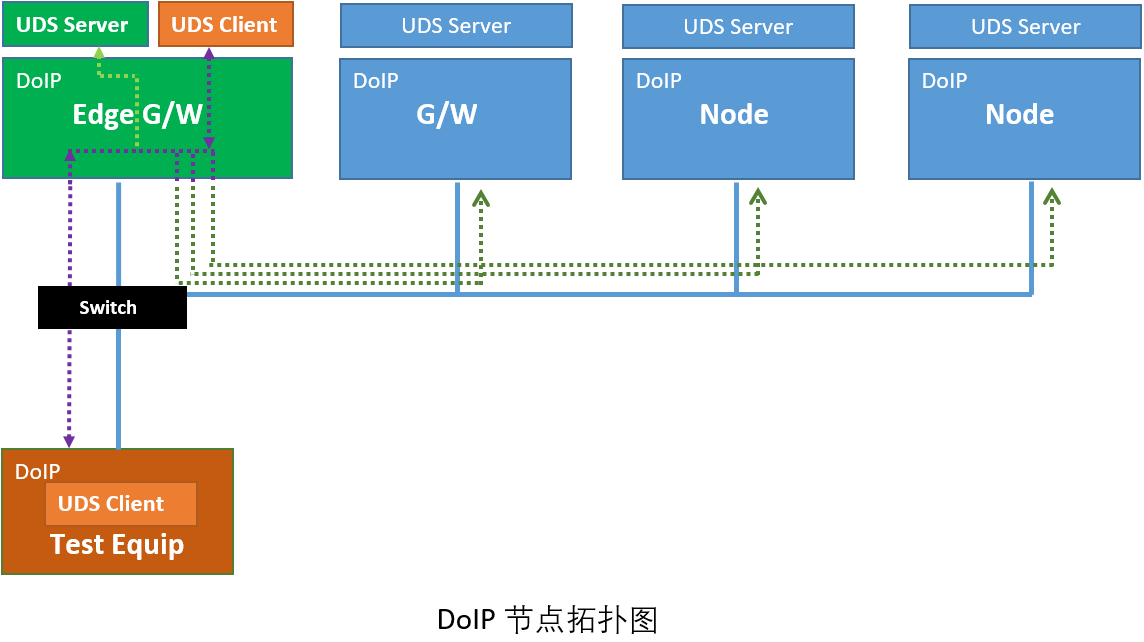
[7.1 ECU异常 41](#_Toc22236216)

[7.2 协议解析异常 43](#_Toc22236217)

[7.3 其他异常 44](#_Toc22236218)

# DoIP工作原理





DoIP协议全称为基于IP网络的诊断通信（Diagnostic communication over Internet Protocol），遵守协议标准ISO 13400。DoIP协议的建立是为了基于TCP/IP协议实现车辆诊断系统的通用要求，使得DoIP诊断相比较传统CAN诊断有着更快的数据诊断响应，从而在复杂的诊断任务和刷新应用的情况下可以极大地节约时间和成本。DoIP诊断经由通用的统一诊断服务UDS协议引入诊断服务，通过传输控制协议TCP，用户数据报协议UDP和以太网协议IP完成测试设备与ECU间的诊断通信。

本DoIP协议栈整体结构由上层的DoIP应用层与底层的DoIP Server端构成，并通过IPC通信完成进程之间的数据通信。其中DoIP应用层提供了两类接口，分别供UDS Client端与UDS Server端使用，底层的DoIP Server由doipdaemon守护进程负责启动。

1.DoIP Client/DoIP Node创建。当Application 启动后，UDS Client需使用doipclient API对DoIP Client实体进行初始化并进行回调函数的注册，回调函数用于接收发送请求的回应消息。UDS Server需使用doipnode API对DoIP Node实体进行初始化并进行回调函数的注册，回调函数用于接收诊断请求。

2.车辆发现及车辆公告。UDS Client端在创建好DoIP Client后，可主动发起车辆发现功能，用以收集DoIP Node的相关信息（VIN、EID、GID、逻辑地址等）。以Edge gateway身份运行的DoIP协议栈（如MPU），在DoIP协议栈的服务端启动并收到网卡分配好IP地址的消息后，通过三次的车辆公告信息广播VIN、EID、GID和逻辑地址，测试设备通过接收广播发送而来的车辆公告也可完成车辆的发现。

3.路由激活。UDS Client端可使用路由激活功能去激活DoIP协议栈，并在收到路由激活成功的响应后，可以对ECU进行诊断。被激活成功的DoIP实体可以对DoIP诊断报文进行路由和处理。

4.诊断请求。UDS Client端可使用诊断请求发送功能去对指定目的地址的ECU进行诊断。UDS Server端可通过注册的诊断请求的回调函数来接收目的地址为自身的诊断请求。并通过诊断请求回应功能回应诊断数据给UDS Client端，此时相应的UDS Client端通过注册的诊断回应的回调函数接收目的地址为自身的诊断回应。

# DoIP Application实现机制

DoIP应用层可分为UDSClient的实现和UDSServer的实现。UDSClient与UDSServer使用的接口文件不同，但使用同一个DoIP客户端的共享链接库，libnedoipstackclient。DoIP应用层根据需求可以同时实现UDSClient / UDSServer，也可以独立实现。

在客户端Application启动之后，主线程会与DoIP协议栈服务端之间建立IPC连接，IPC通信使用的是Unix domain socket。除了主线程，同时还会创建一个监听Socket事件的子线程，用于通过IPC通信来接收DoIP协议栈服务端传输过来的数据。

术语的命名说明。因DoIP标准规范书ISO 13400-2所给的示例时序都是External Test Equipment与DoIP Node之间的交互。运行在External Test Equipment的上层应用是UDS Client，运行在DoIP Node的上层应用是UDS Server。因此，接口的命名对于UDS Client端使用的是xxx\_equip\_xxx，而UDS Server端使用的是xxx\_node\_xxx。

## UDS Client

DoIP 应用层如果需要实现UDS Client的功能，那么应使用ne\_doip\_client.h接口文件。调用形式如ne\_doip\_equip\_xxx函数进行相应xxx服务的请求。调用的请求会封装成内部协议与DoIP协议栈服务端进行IPC通信。DoIP协议栈服务端对收到的IPC通信数据会进行内部协议与标准协议的转义，并将请求路由。

运行UDS Client的DoIP协议栈服务端收到外部UDS Server的回应消息后，会将标准协议转成内部协议，通过IPC通信，传送给上层客户端。此时上层的监听线程读到数据后，进行内部协议的解析，并把相应数据通过ne\_doip\_xxx\_callback回调函数的形式，回调给UDSClient端。回调函数是应用层初始化的过程中，注册进来的。

回调函数得到的数据如果涉及到逻辑处理，需要UDSClient另起线程来处理，以免占用主线程资源。此时，一个发送请求与得到回应的流程结束。

## UDSServer

DoIP 应用层的如果需要实现UDSServer的功能，那么应使用ne\_doip\_node.h接口文件。调用形式如ne\_doip\_node\_xxx函数进行相应xxx服务的请求。调用的请求会封装成内部协议与DoIP协议栈服务端进行IPC通信。DoIP协议栈服务端对收到的IPC通信数据会进行内部协议与标准协议的转义，并将请求回应路由。

运行UDS Server的DoIP协议栈服务端收到外部UDS Client的请求消息后，会将标准协议转成内部协议，通过IPC通信，传送给上层客户端。此时上层的监听线程读到数据后，进行内部协议的解析，并把相应数据通过ne\_doip\_xxx\_callback回调函数的形式，回调给UDS Server端。回调函数是应用层初始化的过程中，注册进来的。

回调函数得到的数据如果涉及到逻辑处理，需要UDSServer另起线程来处理，以免占用主线程资源。UDS Server对解析得到UDS Command内部处理完之后，需要回应诊断请求。此时，调用诊断回应接口将诊断数据回送给UDS Client端，那么一个得到请求与发送回应的流程结束。

## UDS Client/ UDS Server

DoIP应用层可同时实现UDS Client与UDS Server，该场景可实现即作为UDS Client角色去诊断其它ECU或升级其它ECU，也可以作为UDS Server接收被诊断或被升级的需求，还可以作为测试自己诊断自己。实现方式与2.1节和2.2节一样，不同的是，自己诊断自己的情况不会做内部协议与标准协议的转换，而是直接在内部协议之间进行交互。

# DoIP SDK简介

Release 目录列表如下：

Release package

|--bin

|--config

|--demo

|--include

|--lib

## bin目录

bin目录下存放的是DoIP协议栈的服务程序和demo程序bin文件。

doipdaemon：启动并初始化DoIP协议Server端，需要传入DoIP协议栈所需要的外部情报。

uds\_client：DoIP应用层UDS Client端operation用例程序。

uds\_server：DoIP应用层UDS Server端operation用例程序。

## config目录

config目录下存放的是ECU上运行DoIP协议栈所需要的配置文件，文件名为：

doip\_server\_config.xml

因ECU所处的角色不同，使用的配置文件若干字段的设置也不同。角色通常含：“edge gateway”，“gateway”，“node”。config目录下分别存放着各个角色所适用的配置文件。其中“edge gateway”是唯一与外部模块交互的角色，一个局域网内只能有一个“edge gateway”。“gateway”是普通网关，如果挂载了多个ECU则应该使用的是这个角色。“node”是独立的支持以太网的DoIP节点ECU。

## include目录

include目录提供了各ECU的Application开发者使用到DoIP协议栈的客户端接口文件。

## lib目录

lib目录提供了各ECU的APP使用到DoIP协议开发所需要的库文件，以及DoIP协议栈运行所需的库文件。

库的层次依赖状况分布如下：

图片包含 屏幕截图

描述已自动生成

其中Application layer依赖库为Application的开发和运行提供DoIP协议栈客户端的接口，在Application开发编译时需要对该层依赖库进行指定并链接。DoIP Daemon负责加载并启动DoIP协议栈服务端，因此守护进程需要对该层依赖库进行指定并链接。

## demo目录

demo目录下存放调用DoIP 协议栈API接口进行请求、应答、回调等通信的示例源代码，供开发人员参考如何调用相关接口。

# DoIP开发说明

## DoIP协议栈服务端实现说明

doipdaemon作为DoIP协议栈服务端的启动程序，需要初始化协议栈，以及传入相应的外部情报。

### 初始化协议栈

ne\_doip\_server\_init(const char\* config\_path)

参数为配置文件的路径，该初始化所实现的是配置文件的加载，资源的初始化。

### ECU情报传入

ne\_doip\_set\_vin\_info(const char \*vin)

ne\_doip\_set\_eid\_info(const char \*eid)

ne\_doip\_set\_gid\_info(const char \*gid)

doipdaemon负责监听上述情报（VIN/EID/GID）的获取状态，如果得到情报，则透传给DoIP协议栈的服务端，用于车辆发现及车辆公告。

### 使能线激活及非激活

ne\_doip\_activation\_line\_switch\_active()

ne\_doip\_activation\_line\_switch\_deactive()

doipdaemon负责监听使能线的状态，从外部模块得到使能线状态变化的通知后，需要将该通知同时传给DoIP协议栈的服务端，用于判断socket资源的释放与否。

### 有效IP分配

ne\_doip\_request\_ip\_addr\_assignment(const char\* if\_name)

ne\_doip\_release\_ip\_addr\_assignment(const char\* if\_name)

doipdaemon负责监听networkmanager模块对网卡IP是否配置的消息。若配置好IP地址则需要通知DoIP协议栈的服务端，用于开启socket的监听，并启动车辆公告。若IP地址释放，也需要通知DoIP协议栈，清理socket的相应资源。

传入的网卡名需要同配置文件所配置的网卡名一致，用以判断是外部网卡还是内部网卡，以及进行收发数据的网卡绑定。

### ECU Ready状态

ne\_doip\_powermode\_status\_change(unsigned char power\_mode\_status)

doipdaemon负责监听ECU的ready状态，如果状态有变更需要通知DoIP协议栈的服务端。用于是否可诊断的条件判断。0x00：not ready ；0x01：ready。

## DoIP 应用层UDS Client端实现说明

DoIP应用层提供了UDS Client端使用的接口，可以让运行DoIP协议栈的UDSClient与另一运行DoIP协议栈的UDS Server进行交互。

### 初始化与销毁

UDS Client端作为两端DoIP协议栈交互的TestEquipment角色的上层应用，需要首先进行初始化操作，接口使用如下：

ne\_doip\_equip\_create(uint16\_t logical\_source\_address,

ne\_doip\_equip\_callback\_register\_t callback\_register)

传入自身逻辑地址外，还要进行回调函数的注册。callback\_register作为回调函数注册的管理器，提供如车辆发现，路由激活，实体状态请求，power mode请求，诊断相关的回调函数注册。

参考ne\_doip\_comm\_def.h的相关回调函数原型。

当UDS Client退出时，需要调用客户端销毁的接口，使用如下：

ne\_doip\_equip\_destroy(uint16\_t logical\_source\_address)

该接口在退出时，可以释放Test Equipment创建时的相应资源。

### DoIP请求发送

UDS Client端作为DoIP请求的发起方，可以通过调用以下接口形式发送相应的请求

ne\_doip\_equip\_xxx()

xxx为相应请求的类型，如车辆发现请求，路由激活请求，实体状态请求，power mode请求，诊断请求。

每个请求都有相应的回调函数所对应，请求通过DoIP协议栈发送出去并收到了回应后，相应的回调函数将会被调用。

请求的发送请遵循DoIP协议的使用规范

a. DoIP协议栈支持静态配置表（默认）

因静态配置表含有所有ECU的逻辑地址与IP地址的映射关系，所以UDS Client无需进行车辆发现，而直接进行路由激活即可。然后在路由激活成功后，可以对指定目的地址的ECU进行诊断数据的收发工作。

b. DoIP协议栈不支持静态配置表

因不含逻辑地址与IP地址的映射关系，DoIP协议栈不知道网络内其它ECU的信息。所以需要进行车辆发现来获得相应ECU情报（含逻辑地址），同时可得知回应的IP地址。以此来建立逻辑地址与IP地址的映射关系。之后的步骤相同，即需要进行路由激活建立TCP连接，然后发送诊断请求及接受诊断回应。

## DoIP 应用层UDS Server端实现说明

DoIP应用层提供了UDS Server端使用的接口，可以让运行DoIP协议栈的UDS Server与另一运行DoIP协议栈的UDS Client进行交互。

### 初始化与销毁

UDS Server端作为两端DoIP协议栈交互的DoIP Node角色的上层应用，需要首先进行初始化操作，接口使用如下：

ne\_doip\_node\_create(uint16\_t logical\_source\_address,

ne\_doip\_instence\_type\_t doip\_instence\_type,

ne\_doip\_node\_callback\_register\_t callback\_register)

传入自身逻辑地址及实例类型外，还要回调函数的注册。callback\_register作为回调函数注册的管理器，提供两个必选的回调函数注册及一个可选的回调函数（需要定制）注册，详细参考ne\_doip\_comm\_def.h的相关回调函数原型。

当UDS Server退出时，需要调用客户端销毁的接口，使用如下：

ne\_doip\_node\_destroy (uint16\_t logical\_source\_address)

该接口在退出时，可以释放创建时对应逻辑地址的DoIPNode的资源。因可创建多个DoIPNode节点，所以在销毁最后一个DoIPNode时，才会释放最初创建的共通资源。

### DoIP请求回应

UDS Server端作为DoIP请求的接收方，当DoIP协议栈解析出 UDS data后，通过回调的形式转送给UDS Server。UDS Server经过内部的逻辑处理后，需要回应诊断数据给UDS Client，调用的接口如下：

ne\_doip\_node\_diag\_data\_request (ne\_doip\_diag\_data\_request\_t\* diag\_data\_request)

diag\_data\_request封装了UDS data等信息，详见ne\_doip\_comm\_def.h中的定义。通过该接口将诊断回应的数据通过IPC通信传达到DoIP协议栈的服务端。服务端利用逻辑地址与IP地址的映射关系，会正确送达到对应UDS Client的DoIP协议栈端。

# DoIP安装部署

## 角色说明

三种角色“edge\_gateway”，“gateway”，“node”的配置文件均在./config下。一台设备只能有一个身份，运行时需要拷贝配置文件到/tmp目录下。

配置文件包括了网卡名，路由表等信息。在routing标签下的entity ID=”0”的一项，ip标签应该设为127.0.0.1，代表本机IP地址，logical address标签的第一条为本机逻辑地址。一个entity ID下的多条逻辑地址，代表该entity上挂载多个doip node的逻辑地址。

1. MPU角色是“edge\_gateway”，其配置文件存放在了如下目录中

./config/edge\_gateway\_config/doip\_server\_config.xml。

本例中MPU 实际分配IP为:192.168.29.1，logical addr:0x0201，ifname:enp1s0。

1. MCU角色是“gateway”，其配置文件存放在了如下目录中

./config/gateway\_config/doip\_server\_config.xml。

本例中MCU 实际分配IP为:192.168.29.33，logical addr:0x0301，ifname:enp1s0。假设MCU下挂载一个node，对应逻辑地址为0x0302，对于该node的诊断由MCU转发。

1. 以太网ECU角色是“node”，其配置文件存放在了如下目录中

./config/node\_config /doip\_server\_config.xml。

本例中ECU实际分配IP为:192.168.29.36，logical addr:0x0130。

## 程序部署说明

1. 以两个设备作为测试，一台作为MPU，一台作为以太网ECU。

【MPU】 IP addr: 192.168.29.1，logical addr: 0x0201，网卡接口ifname: enp1s0；运行doipdaemon和udsclient作为UDS的client端。

【以太网ECU】 IP addr: 192.168.29.36，logical addr: 0x0130，网卡接口ifname: enp1s0；运行doipdaemon和udsserver作为UDS的server端。

2. 程序运行顺序：

MPU的doipdaemon

以太网ECU的doipdaemon

以太网ECU的udsserver

MPU的udsclient

3. doipdaemon的配置，在./demo/diagd/main.c中，有手动设置VIN EID GID的函数，需要保证MPU和ECU的VIN相同，EID不同（通常使用MAC地址）。GID与EID保持一致。

## 程序运行

### 网络配置

1. 确保MPU与以太网ECU处于同一网段，确保两台设备可以通信。本例子使用的是192.168.29.0/24网段。

2. 修改MPU的网络配置。修改./config/edge\_gateway\_config/doip\_server\_config.xml文件，确保本机IP addr和logical addr正确，本例中MPU IP addr:192.168.29.1，logical addr:0x0201。

3. 修改以太网ECU网络配置。修改./config/node\_config/doip\_server\_config.xml文件，确保本机IP addr和logical addr正确，本例中ECU IP addr:192.168.29.36，logical addr: 0x0130。

### 启动uds server（以太网ECU）

1. 确保配置文件./config/node\_config/doip\_server\_config.xml内容正确无误。拷贝以太网ECU配置文件到/tmp目录。运行：

cp ./config/node\_config/doip\_server\_config.xml /tmp

2. 确保处在doip\_release目录下，切换到root：

su root

3.点运行. envsetup.sh设置运行库地址。也可手动设置运行库地址：

. envsetup.sh或

LD\_LIBRARY\_PATH=./lib

4. 若修改过demo下的代码，如修改EID GID，修改逻辑地址，修改设备身份等等操作，需要重新编译。若点运行过envsetup.sh可以使用envsetup.sh中的makefile脚本。

5. 先运行doipdaemon后运行udsserver

./bin/doipdaemon &

./bin/udsserver &

6. 由于doipdaemon和udsserver的程序日志参杂到一起容易混淆，所以可以将doipdaemon和udsserver放在两个终端运行。同样的每个终端必须使用root权限并配置运行库地址。

### 启动uds client（MPU）

1. 确保配置文件./config/edge\_gateway\_config/doip\_server\_config.xml内容正确无误。拷贝MPU配置文件到/tmp目录。运行：

cp ./config/edge\_gateway\_config/doip\_server\_config.xml /tmp

2. 确保处在doip\_release目录下，切换到root：

su root

3.点运行. envsetup.sh设置运行库地址。也可手动设置运行库地址：

. envsetup.sh或

LD\_LIBRARY\_PATH=./lib

4. 若修改过demo下的代码，如修改EID GID，修改逻辑地址，修改设备身份等等操作，需要重新编译。若点运行过envsetup.sh可以使用envsetup.sh中的makefile脚本。

5. 先运行doipdaemon后运行udsclient

./bin/doipdaemon &

./bin/udsclient &

6. 由于doipdaemon和udsclient的程序日志参杂到一起容易混淆，所以可以将doipdaemon和udsclient放在两个终端运行。同样的每个终端必须使用root权限并配置运行库地址。

## 程序日志分析

### MPU端的udsclient日志（./bin/udsclient）

1. **创建doip equip**

[10-15 11:39:12.73909 line:0164] ne\_doip\_equip\_create

[10-15 11:39:12.73947 line:0024] [test equip] ne\_doip\_pack\_test\_equip\_regist start ..

1. **车辆发现请求及回应**

[10-15 11:39:13.73982 line:0047] [test equip] ne\_doip\_pack\_vehicle\_identify start ..

[10-15 11:39:13.74070 line:0459] [test equip] ne\_doip\_equip\_unpack start ..

recv data is F22000000049564938384142434431393830303431380201180373CB6B33180373CB6B3300

[10-15 11:39:13.74081 line:0291] internal payload type is [NE\_DOIP\_IN\_EQUIP\_ANN\_IDEN\_RES]!

equip\_vehicle\_identity\_callback\_func is enter...

ne\_doip\_vehicle\_identity\_resinfo\_t vin is 4956493838414243443139383030343138

ne\_doip\_vehicle\_identity\_resinfo\_t eid is 180373CB6B33

ne\_doip\_vehicle\_identity\_resinfo\_t gid is 180373CB6B33

ne\_doip\_vehicle\_identity\_resinfo\_t logical\_address is 0201

1. **路由激活请求及回应**

[10-15 11:39:13.74094 line:0110] [test equip] ne\_doip\_pack\_routing\_active start ..

[10-15 11:39:13.74162 line:0459] [test equip] ne\_doip\_equip\_unpack start ..

recv data is F609000000020102011000000000

[10-15 11:39:13.74169 line:0319] internal payload type is [NE\_DOIP\_IN\_EQUIP\_ROUTING\_ACTIVE]!

equip\_routing\_active\_callback\_func is enter...

ne\_doip\_routing\_active\_resinfo\_t logical\_address is 0201

ne\_doip\_routing\_active\_resinfo\_t routing\_active\_res\_code is 16

1. **诊断请求及回应**

[10-15 11:39:13.74187 line:0229] [test equip] ne\_doip\_pack\_diagnositc\_req start ..

[10-15 11:39:13.74566 line:0459] [test equip] ne\_doip\_equip\_unpack start ..

recv data is FA050000000130020100

[10-15 11:39:13.74571 line:0381] internal payload type is [NE\_DOIP\_IN\_EQUIP\_DIAGNOSTIC\_PACK]!

equip\_diagnostic\_pack\_callback\_func is enter...

[10-15 11:39:14.74710 line:0459] [test equip] ne\_doip\_equip\_unpack start ..

recv data is FC0A00000001300201010203040506

[10-15 11:39:14.74714 line:0427] internal payload type is [NE\_DOIP\_IN\_EQUIP\_DIAGNOSTIC]!

equip\_diagnostic\_callback\_func is enter...

equipment logical\_target\_address is [0130]

recv data is 010203040506

1. **entity状态请求及回应**

[10-15 11:39:14.74718 line:0176] [test equip] ne\_doip\_pack\_entity\_status start ..

[10-15 11:39:14.74745 line:0459] [test equip] ne\_doip\_equip\_unpack start ..

recv data is F80700000000FF00001EE240

[10-15 11:39:14.74749 line:0353] internal payload type is [NE\_DOIP\_IN\_EQUIP\_ENTITY\_STATUS]!

equip\_entity\_status\_callback\_func is enter...

ne\_doip\_entity\_status\_info\_t node\_type is 0

ne\_doip\_entity\_status\_info\_t MCTS is 255

ne\_doip\_entity\_status\_info\_t NCTS is 0

ne\_doip\_entity\_status\_info\_t MDS is 2024000

1. **power mode请求及回应**

[10-15 11:39:14.74754 line:0200] [test equip] ne\_doip\_pack\_power\_mode start ..

[10-15 11:39:14.74778 line:0459] [test equip] ne\_doip\_equip\_unpack start ..

recv data is F90100000001

[10-15 11:39:14.74782 line:0373] internal payload type is [NE\_DOIP\_IN\_EQUIP\_POWER\_MODE]!

equip\_power\_mode\_callback\_func is enter...

power\_mode is 1

### ECU端的udsserver日志（./bin/udsserver）

1. **Doip节点创建**

[10-15 11:38:20.02571 line:0180] ne\_doip\_node\_create is start..[0130]

[10-15 11:38:20.02613 line:0025] [doip\_node] ne\_doip\_pack\_announce start ..

[10-15 11:38:20.02635 line:0258] ne\_doip\_node\_create is end..[0130]

1. **收到indication**

[10-15 11:39:07.29669 line:0362] [doip\_node] ne\_doip\_node\_unpack start ..

recv data is 040A0000000002010130112233332211

[10-15 11:39:07.29675 line:0213] type is [NE\_DOIP\_IN\_PAYLOADTYPE\_DIAG\_INDICATION] .

ne\_uds\_server\_diag\_indication\_callback is enter...

diag\_indication\_callback is called... result code is [0]

logical\_source\_address is [0201]

logical\_target\_address is [0130]

TAType is [0]

1. **回应诊断请求**

ne\_uds\_server\_send\_data is enter...

[10-15 11:39:08.29721 line:0118] ne\_doip\_pack\_diagnositc\_res start ..

1. **收到rsp\_confirm**

[10-15 11:39:08.29819 line:0362] [doip\_node] ne\_doip\_node\_unpack start ..

recv data is 030400000002010130

[10-15 11:39:08.29825 line:0317] type is [NE\_DOIP\_IN\_PAYLOADTYPE\_DIAG\_CONFIRM] .

ne\_uds\_server\_diag\_confirm\_callback is enter...

diag\_confirm\_callback\_1 is called... result code is [0]

logical\_source\_address is [0201]

logical\_target\_address is [0130]

TAType is [0]

### MPU端的doipdaemon （./bin/doipdaemon）

1. **初始化**

[10-15 11:39:02.68217 line:1707] ne\_doip\_server\_init enter!

[10-15 11:39:02.68366 line:0280] ne\_doip\_server\_manager\_create is start.

[10-15 11:39:02.68387 line:0349] ne\_doip\_server\_manager\_create is end.

[10-15 11:39:02.68399 line:1774] ne\_doip\_set\_vin\_info is enter. vin is IVI88ABCD19800418

[10-15 11:39:02.68401 line:1788] ne\_doip\_set\_eid\_info is enter. eid is 180373CB6B33

[10-15 11:39:02.68419 line:1801] ne\_doip\_set\_gid\_info is enter. gid is 180373CB6B33

[10-15 11:39:03.68479 line:1816] ne\_doip\_activation\_line\_switch\_active enter!

[10-15 11:39:04.68521 line:1841] ne\_doip\_request\_ip\_addr\_assignment enter! ifname is enp1s0

[10-15 11:39:04.68551 line:1171] ipv4\_udp\_socket\_external bind to if:enp1s0

[10-15 11:39:04.68554 line:1177] ipv4\_udp\_socket\_external bind to port:58152

1. **开始车辆公告，忽略自身的车辆公告**

[10-15 11:39:04.68564 line:1901] ne\_doip\_request\_ip\_addr\_assignment is enter start vehicle\_announce

[10-15 11:39:05.06053 line:4676] ne\_doip\_vehicle\_announce\_wait\_timer\_callback is start...timeid[1]

[10-15 11:39:05.06062 line:0669] ne\_doip\_pack\_announce\_or\_identityresponse start...

[10-15 11:39:05.06066 line:3763] [ne\_doip\_pack] start....

[length: 40]pack data is 02FD00040000002049564938384142434431393830303431380201180373CB6B33180373CB6B3300

[10-15 11:39:05.06112 line:0685] udp socketfd is [9].ip is [192.168.29.1], port is [58152]

[10-15 11:39:05.06116 line:4234] [ne\_doip\_net\_unpack] start....

recv data is 02FD00040000002049564938384142434431393830303431380201180373CB6B33180373CB6B3300

[10-15 11:39:05.06123 line:4031] [ne\_doip\_net\_unpack\_exec] payload\_length is 32 byte

[10-15 11:39:05.06126 line:4147] [ne\_doip\_net\_unpack\_exec] This is an internal broadcast message, ignored

[10-15 11:39:05.56150 line:4717] ne\_doip\_vehicle\_announce\_interval\_timer\_callback is start...timeid[2]

[10-15 11:39:05.56159 line:0669] ne\_doip\_pack\_announce\_or\_identityresponse start...

[10-15 11:39:05.56162 line:3763] [ne\_doip\_pack] start....

[length: 40]pack data is 02FD00040000002049564938384142434431393830303431380201180373CB6B33180373CB6B3300

[10-15 11:39:05.56213 line:0685] udp socketfd is [9].ip is [192.168.29.1], port is [58152]

[10-15 11:39:05.56218 line:4234] [ne\_doip\_net\_unpack] start....

recv data is 02FD00040000002049564938384142434431393830303431380201180373CB6B33180373CB6B3300

[10-15 11:39:05.56226 line:4031] [ne\_doip\_net\_unpack\_exec] payload\_length is 32 byte

[10-15 11:39:05.56229 line:4147] [ne\_doip\_net\_unpack\_exec] This is an internal broadcast message, ignored

[10-15 11:39:06.06151 line:4717] ne\_doip\_vehicle\_announce\_interval\_timer\_callback is start...timeid[2]

[10-15 11:39:06.06160 line:0669] ne\_doip\_pack\_announce\_or\_identityresponse start...

[10-15 11:39:06.06163 line:3763] [ne\_doip\_pack] start....

[length: 40]pack data is 02FD00040000002049564938384142434431393830303431380201180373CB6B33180373CB6B3300

[10-15 11:39:06.06212 line:0685] udp socketfd is [9].ip is [192.168.29.1], port is [58152]

[10-15 11:39:06.06218 line:4234] [ne\_doip\_net\_unpack] start....

recv data is 02FD00040000002049564938384142434431393830303431380201180373CB6B33180373CB6B3300

[10-15 11:39:06.06225 line:4031] [ne\_doip\_net\_unpack\_exec] payload\_length is 32 byte

[10-15 11:39:06.06229 line:4147] [ne\_doip\_net\_unpack\_exec] This is an internal broadcast message, ignored

1. **MPU的uds client初始化**

[10-15 11:39:12.73949 line:0416] ne\_doip\_ipc\_socket\_data enter

[10-15 11:39:12.73957 line:0434] ne\_doip\_ipc\_socket\_data accept fd:11, sun\_path:

[10-15 11:39:12.73971 line:0447] ne\_doip\_ipc\_socket\_data ipc\_client\_list length:[0]

[10-15 11:39:12.74015 line:4571] [ne\_doip\_ipc\_unpack] start....

recv data is F0020000000201

[10-15 11:39:12.74020 line:4446] request\_type is NE\_DOIP\_IN\_EQUIP\_RGIST

[10-15 11:39:12.74023 line:4465] registed equip address is 0201

1. **uds client发送车辆发现**

[10-15 11:39:13.74028 line:4571] [ne\_doip\_ipc\_unpack] start....

recv data is F300000000

[10-15 11:39:13.74032 line:4470] request\_type is NE\_DOIP\_IN\_EQUIP\_IDENTITY\_REQ

[10-15 11:39:13.74036 line:0669] ne\_doip\_pack\_announce\_or\_identityresponse start...

[10-15 11:39:13.74040 line:0715] vehicle identity response to internal test equipemt...

[10-15 11:39:13.74053 line:3763] [ne\_doip\_pack] start....

[length: 8]pack data is 02FD000100000000

[10-15 11:39:13.74094 line:0685] udp socketfd is [9].ip is [192.168.29.1], port is [58152]

[10-15 11:39:13.74099 line:4234] [ne\_doip\_net\_unpack] start....

recv data is 02FD000100000000

[10-15 11:39:13.74109 line:4031] [ne\_doip\_net\_unpack\_exec] payload\_length is 0 byte

[10-15 11:39:13.74115 line:4049] [ne\_doip\_net\_unpack\_exec] payload\_type is [Vehicle Identification request message]

[10-15 11:39:13.74126 line:4057] [ne\_doip\_net\_unpack\_exec] This is an internal broadcast message, ignored

1. **uds client发送路由激活**

[10-15 11:39:13.74130 line:4571] [ne\_doip\_ipc\_unpack] start....

recv data is F60D00000000000000000002010000000000

[10-15 11:39:13.74137 line:4488] request\_type is NE\_DOIP\_IN\_EQUIP\_ROUTING\_ACTIVE

[10-15 11:39:13.74141 line:1010] routing activation doip entity of self network..

[10-15 11:39:13.74148 line:0977] routing activation response to internal test equipemt...

uds client发送诊断请求

[10-15 11:39:13.74211 line:4571] [ne\_doip\_ipc\_unpack] start....

recv data is FC0A00000000000002010130112233332211

[10-15 11:39:13.74216 line:4512] request\_type is NE\_DOIP\_IN\_EQUIP\_DIAGNOSTIC

[10-15 11:39:13.74219 line:1970] [id:0]send diag\_data\_total\_length is [10 byte], diag\_data\_current\_pos is [10 byte]..

[10-15 11:39:13.74223 line:2043] Edge gatewey routing start..to [0130]..

[10-15 11:39:13.74226 line:1438] ne\_doip\_test\_tcp\_create enter! ip is [192.168.29.36], port is [13400]

[10-15 11:39:13.74275 line:0273] ne\_doip\_select\_add fd is [12]

[10-15 11:39:13.74290 line:1686] create test tcp socket fd is [12]...

[10-15 11:39:13.74294 line:1719] Cache data and routing activation.

[10-15 11:39:13.74305 line:0506] ne\_doip\_add\_connection\_table is enter...fd is [12] comm\_type is [3]

[10-15 11:39:13.74311 line:0033] queue init is succeed !

[10-15 11:39:13.74315 line:0099] insert first data is succeed !

[10-15 11:39:13.74318 line:3763] [ne\_doip\_pack] start....

[length: 15]pack data is 02FD00050000000702010000000000

1. **路由激活回应**

[10-15 11:39:13.74419 line:4234] [ne\_doip\_net\_unpack] start....

recv data is 02FD000600000009020101301000000000

[10-15 11:39:13.74424 line:4031] [ne\_doip\_net\_unpack\_exec] payload\_length is 9 byte

[10-15 11:39:13.74427 line:4173] [ne\_doip\_net\_unpack\_exec] payload\_type is [Routing Actiocation response]

[10-15 11:39:13.74430 line:3413] routing activation res enter...res code is [16]..

[10-15 11:39:13.74433 line:0132] queue front data is succeed !

[10-15 11:39:13.74436 line:3763] [ne\_doip\_pack] start....

[length: 18]pack data is 02FD80010000000A02010130112233332211

[10-15 11:39:13.74446 line:0055] ne\_doip\_connection\_write num:[18], fd:[12]

1. **诊断确认req\_confirm**

[10-15 11:39:13.74542 line:4234] [ne\_doip\_net\_unpack] start....

recv data is 02FD8002000000050130020100

[10-15 11:39:13.74547 line:4031] [ne\_doip\_net\_unpack\_exec] payload\_length is 5 byte

[10-15 11:39:13.74550 line:4191] [ne\_doip\_net\_unpack\_exec] payload\_type is [Diagnostic positive acknowledge]

1. **收到诊断响应**

[10-15 11:39:14.74692 line:4234] [ne\_doip\_net\_unpack] start....

recv data is 02FD80010000000A01300201010203040506

[10-15 11:39:14.74696 line:4031] [ne\_doip\_net\_unpack\_exec] payload\_length is 10 byte

[10-15 11:39:14.74698 line:4111] [ne\_doip\_net\_unpack\_exec] payload\_type is [Diagnostic message]

1. **uds client发送 entity状态请求**

[10-15 11:39:14.74733 line:4571] [ne\_doip\_ipc\_unpack] start....

recv data is F806000000180373CB6B33

[10-15 11:39:14.74736 line:4500] request\_type is NE\_DOIP\_IN\_EQUIP\_ENTITY\_STATUS

[10-15 11:39:14.74737 line:1260] entity status response to internal test equipemt...

1. **uds client发送power mode请求**

[10-15 11:39:14.74767 line:4571] [ne\_doip\_ipc\_unpack] start....

recv data is F906000000180373CB6B33

[10-15 11:39:14.74770 line:4506] request\_type is NE\_DOIP\_IN\_EQUIP\_POWER\_MODE

[10-15 11:39:14.74771 line:1338] power mode response to internal test equipemt...

### ECU端的doipdaemon （./bin/doipdaemon）

1. **doip初始化**

[10-15 11:38:10.23321 line:1707] ne\_doip\_server\_init enter!

[10-15 11:38:10.23499 line:0280] ne\_doip\_server\_manager\_create is start.

[10-15 11:38:10.23522 line:0349] ne\_doip\_server\_manager\_create is end.

[10-15 11:38:10.23568 line:1774] ne\_doip\_set\_vin\_info is enter. vin is IVI88ABCD19800418

[10-15 11:38:10.23570 line:1788] ne\_doip\_set\_eid\_info is enter. eid is 180373CB6B34

[10-15 11:38:10.23572 line:1801] ne\_doip\_set\_gid\_info is enter. gid is 180373CB6B34

[10-15 11:38:12.23671 line:1841] ne\_doip\_request\_ip\_addr\_assignment enter! ifname is enp1s0

[10-15 11:38:12.23718 line:1171] ipv4\_udp\_socket\_external bind to if:enp1s0

[10-15 11:38:12.23722 line:1177] ipv4\_udp\_socket\_external bind to port:55141

1. **uds server初始化**

[10-15 11:38:20.02609 line:0416] ne\_doip\_ipc\_socket\_data enter

[10-15 11:38:20.02615 line:0434] ne\_doip\_ipc\_socket\_data accept fd:11, sun\_path:

[10-15 11:38:20.02626 line:0447] ne\_doip\_ipc\_socket\_data ipc\_client\_list length:[0]

[10-15 11:38:20.02663 line:0088] ne\_doip\_connection\_read num:[8], fd:11

[10-15 11:38:20.02667 line:4571] [ne\_doip\_ipc\_unpack] start....

recv data is 0103000000013000

[10-15 11:38:20.02672 line:4334] request\_type is NE\_DOIP\_IN\_PAYLOADTYPE\_NODE\_RGIST

[10-15 11:38:20.02676 line:4358] registed node address is 0130

[10-15 11:39:07.29275 line:0120] ne\_doip\_connection\_recvfrom num:[8]

[10-15 11:39:07.29278 line:0685] udp socketfd is [9].ip is [192.168.29.1], port is [58152]

[10-15 11:39:07.29281 line:4234] [ne\_doip\_net\_unpack] start....

recv data is 02FD000100000000

[10-15 11:39:07.29285 line:4031] [ne\_doip\_net\_unpack\_exec] payload\_length is 0 byte

[10-15 11:39:07.29455 line:0501] ne\_doip\_tcp\_socket\_data accept fd:[12], addr:[192.168.29.1], port:[33176]

[10-15 11:39:07.29461 line:0213] ne\_doip\_os\_get\_ifname is enter. fd[12], ip:[192.168.29.36].

[10-15 11:39:07.29467 line:0521] finded ifname is [enp1s0].

[10-15 11:39:07.29470 line:0273] ne\_doip\_select\_add fd is [12]

[10-15 11:39:07.29480 line:0506] ne\_doip\_add\_connection\_table is enter...fd is [12] comm\_type is [2]

1. **收到路由激活**

[10-15 11:39:07.29507 line:4234] [ne\_doip\_net\_unpack] start....

recv data is 02FD00050000000702010000000000

[10-15 11:39:07.29513 line:4031] [ne\_doip\_net\_unpack\_exec] payload\_length is 7 byte

[10-15 11:39:07.29516 line:4087] [ne\_doip\_net\_unpack\_exec] payload\_type is [Routing activation request] [fd:12]

[10-15 11:39:07.29527 line:2759] [ne\_doip\_unpack\_routing\_activation\_req] equip\_logical\_address is 0201

[10-15 11:39:07.29530 line:2760] [ne\_doip\_unpack\_routing\_activation\_req] activation\_type is 00

[10-15 11:39:07.29532 line:2761] [ne\_doip\_unpack\_routing\_activation\_req] ra\_reserved1 is 00000000

[10-15 11:39:07.29535 line:2799] number of registered tcp socket is 0..

[10-15 11:39:07.29538 line:2641] ne\_doip\_routing\_activation\_check\_authentication is enter.

[10-15 11:39:07.29540 line:2592] ne\_doip\_routing\_activation\_check\_confirmation is enter.

[10-15 11:39:07.29543 line:3763] [ne\_doip\_pack] start....

[length: 17]pack data is 02FD000600000009020101301000000000

1. **收到诊断请求**

[10-15 11:39:07.29632 line:4234] [ne\_doip\_net\_unpack] start....

recv data is 02FD80010000000A02010130112233332211

[10-15 11:39:07.29638 line:4031] [ne\_doip\_net\_unpack\_exec] payload\_length is 10 byte

[10-15 11:39:07.29641 line:4111] [ne\_doip\_net\_unpack\_exec] payload\_type is [Diagnostic message]

[10-15 11:39:07.29644 line:3080] [12: ]recv diag\_data\_total\_length is [10 byte], diag\_data\_current\_pos is [10 byte]..

[10-15 11:39:07.29653 line:2219] ne\_doip\_pack\_diagnostic\_positive\_ack is enter...

[10-15 11:39:07.29658 line:3763] [ne\_doip\_pack] start....

[length: 13]pack data is 02FD8002000000050130020100

1. **诊断请求回应**

[10-15 11:39:08.29755 line:4571] [ne\_doip\_ipc\_unpack] start....

recv data is 020A00000001300201010203040506

[10-15 11:39:08.29760 line:4440] request\_type is NE\_DOIP\_IN\_PAYLOADTYPE\_DIAG\_REQUEST

[10-15 11:39:08.29763 line:3763] [ne\_doip\_pack] start....

[length: 18]pack data is 02FD80010000000A01300201010203040506

## 注意事项

1. 保证设备网络ip addr配置在同一网段，保证设备网络畅通。

2. 在demo/diagd/main.c中可以修改配置文件位置和ECU情报（EID，GID，VIN）等。需要保证代码中的配置文件地址和本机的配置文件地址相同，并且角色对应无误。

3. ECU，MPU，MCU的VIN应相同，EID应均不同，GID应等于EID。

4. 配置文件中的网卡名需要与本机网卡名相同，enp1s0或者eth0或者其他名字需要根据具体情况修改配置文件。

5. 配置文件，特别是“edge\_gateway”的配置文件，需要有正确的路由表，及正确的网络拓扑和ip addr。

6. 配置文件，entity ID=”0”的第一条logical addr是本机逻辑地址。

7. 确保在doip\_release下运行. envsetup.sh再开始运行程序，因为该脚本配置了运行库地址。切换用户到root或者切换回普通用户都需要重新运行该脚本。如果当前目录不是doip\_release则会导致lib目录配置错误。可以参照该脚本手动指定运行库目录。

8. 如果修改了demo代码，则需要重新编译文件，使用envsetup.sh中的makefile能一键编译。要修改的情况有：

修改ECU情报

修改配置文件地址

修改uds server或者uds client的逻辑地址

修改角色，将uds server运行于MCU上。若需要，则需要再uds server中注册两个逻辑地址，并配置正确的配置文件（包括“edge\_gateway”的配置文件，因“edge\_gateway”统筹所有的逻辑地址与IP地址的映射关系）。

9. 必须使用root权限运行。

10. 一台机器中不能启动两个doipdaemon。

# DoIP抓包与报文分析

在定位以太网通信问题时，常常需要通过抓包工具（wireshark, tcpdump）抓包，然后再对报文分析，从而定位和排查问题。对于DoIP中问题的定位，也是通过同样的方式来进行。

## 通过 wireshark 工具进行抓包



Wireshark 的 Filter 内输入如下一些过滤条件,可以更加方便地分析数据来源:

tcp.port== 80 //过滤来自 80 端口的 TCP 数据

udp.port== 12345 //过滤来自 12345 端口的 UDP 数据

ip.src== 192.168.0.1 //过滤源 IP 为 192.168.0.1 的数据

ip.dst== 192.168.0.1 //过目的 IP 为 192.168.0.1 的数据

以上过滤条件可以用 and 跟 or 相互组合,例如 :

tcp.port== 80 and ip.src == 192.168.0.1 //过滤来自 80 端口,源 IP 为 192.168.0.1 的 TCP 数据

udp.port== 12345 or ip.dst == 192.168.0.1 //过滤来自 12345 端口的 UDP 数据,或者目的 IP

为 192.168.0.1 的数据

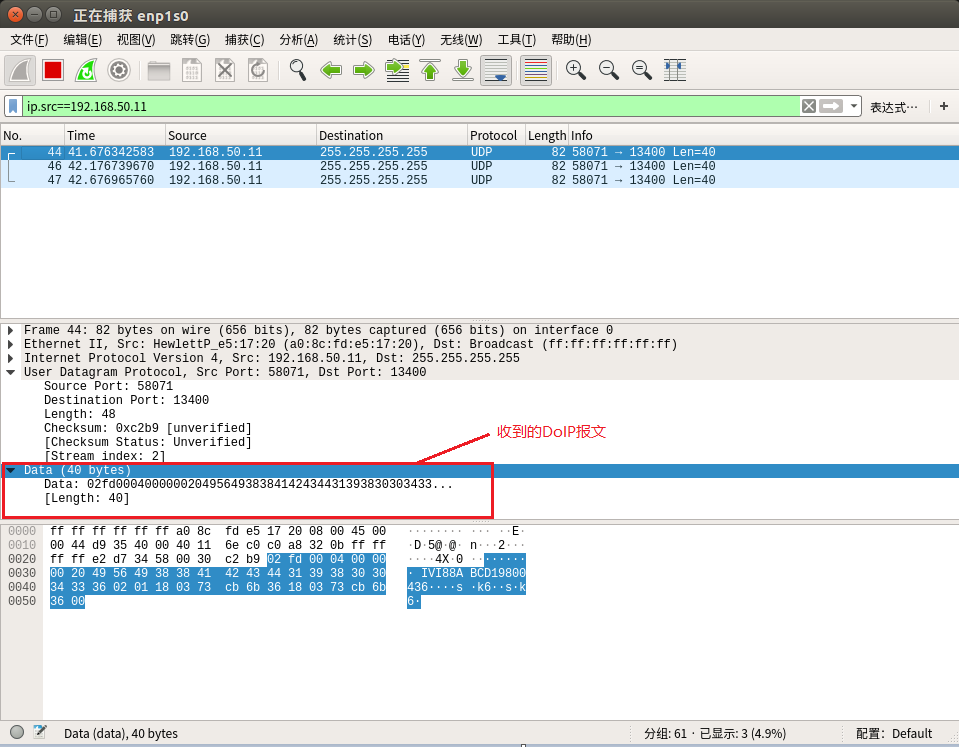
## DoIP报文分析

DoIP主要机能有车辆公告、车辆发现、基于VIN的车辆发现、基于EID的车辆发现、powermode查询、entity状态查询、路由激活、诊断请求。下面将对各个机能进行抓包分析。

### DoIP车辆公告报文分析

DoIP协议栈服务端启动的时候，会发送三次车辆公告，源地址就是DoIP协议栈的源地址(目前为192.168.50.11)，目标地址是广播组”255.255.255.255”。

模拟的外部测试设备收到的报文如下：



edgegateway发送车辆公告，外部测试设备程序收到了三次车辆公告的UDP信息：

具体的DoIP报文分析如下：

字节段02代表：Protocol version

字节段fd代表：Inverse protocol version

字节段0004代表：Vehicle announcement message/vehicle identification response message

字节段00000020代表：Payload length(DoIP协议头之后的数据大小)

之后的字节代表车辆公告实际的数据：也就是vin、gid、eid的实际值。

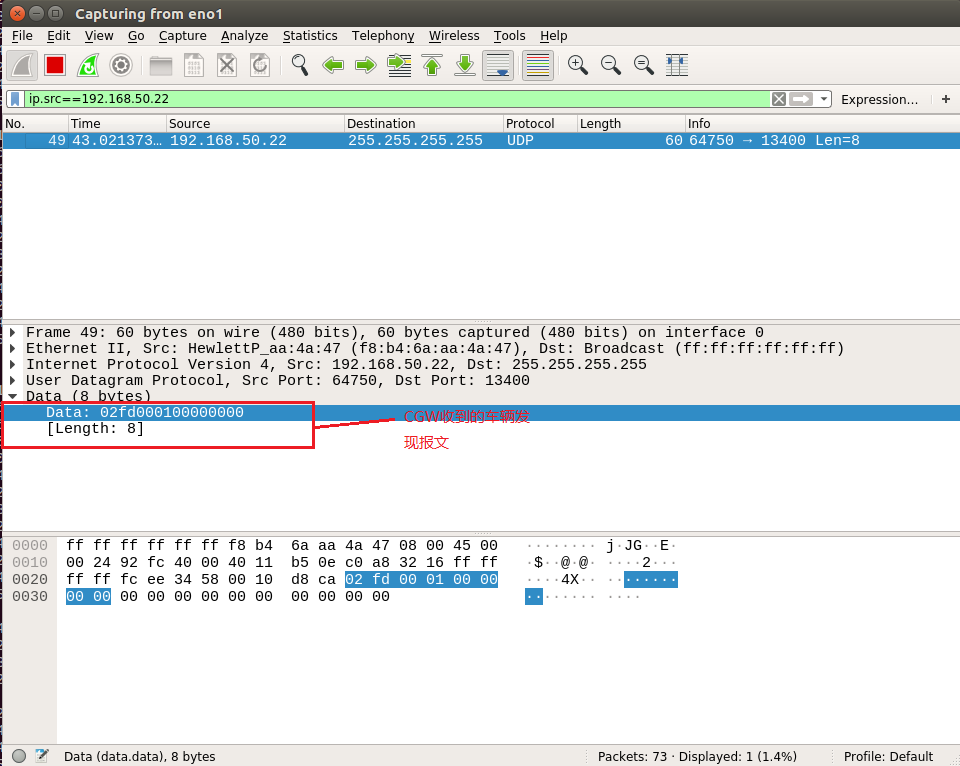
### DoIP车辆发现报文分析

如果外部测试设备启动的比较慢，没有收到车辆公告的信息，则外部测试设备可以主动发送车辆发现请求，获取需要的车辆信息。

发送车辆发现请求方：外部测试设备

接收并回应方：edgegateway(如MPU)

edgegateway收到的UDP报文如下图所示：



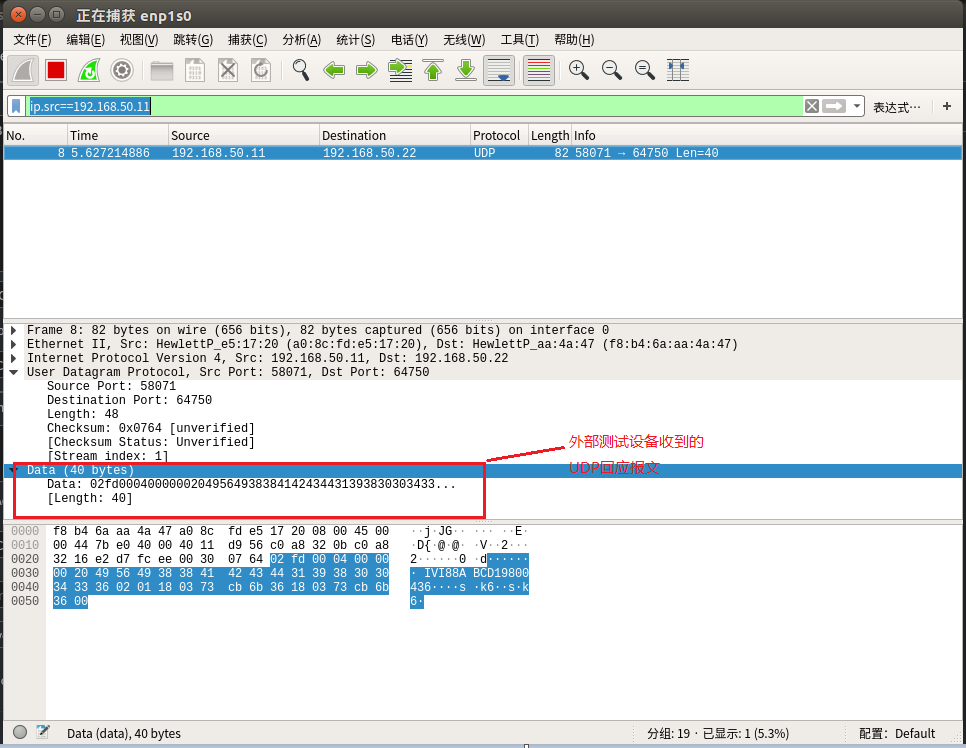
字节段02代表：Protocol version

字节段fd代表：Inverse protocol version

字节段0001代表：Vehicle identification request message

字节段00000000代表：Payload length(DoIP协议头之后的数据大小)

edgegateway回应消息给外部测试设备的UDP报文如下图所示(外部测试设备收到的UDP报文)：



字节段02代表：Protocol version

字节段fd代表：Inverse protocol version

字节段0004代表：Vehicle announcement message/vehicle identification response message

字节段00000020代表：Payload length(DoIP协议头之后的数据大小)

之后的字节代表车辆发现回应的数据：也就是vin、gid、eid数据信息。

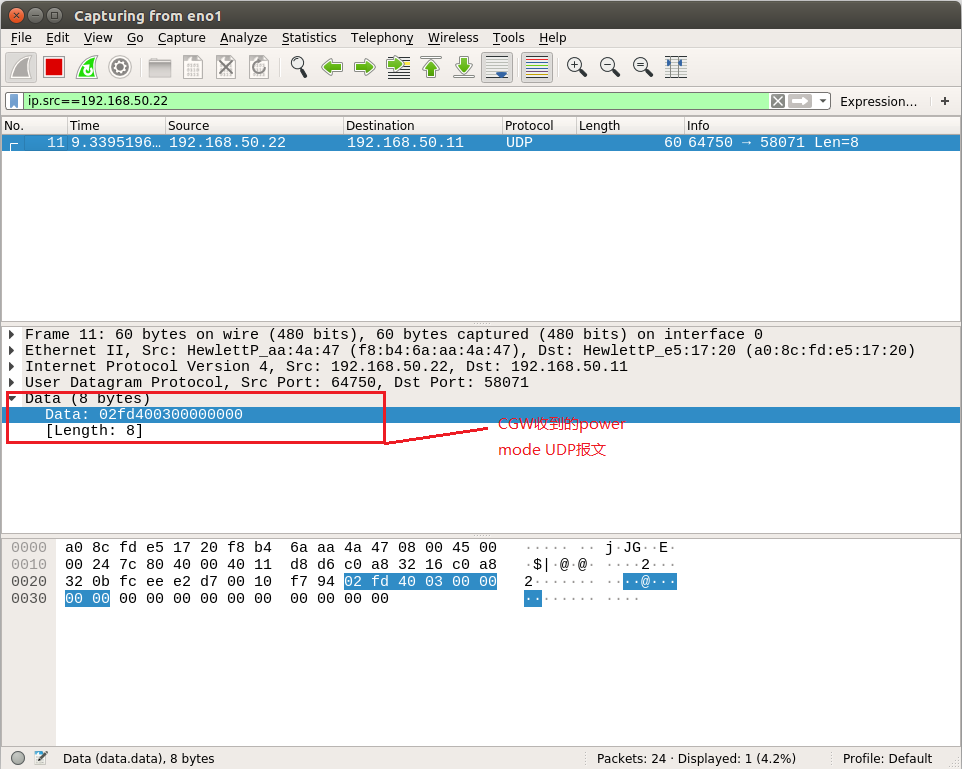
### DoIPPowermode报文分析

Powermode用于查询ECU是否处于可诊断的状态。

发送powermode请求方：外部测试设备

接收并回应方：edgegateway(如MPU)

edgegateway收到的powermodeUDP报文如下图所示：



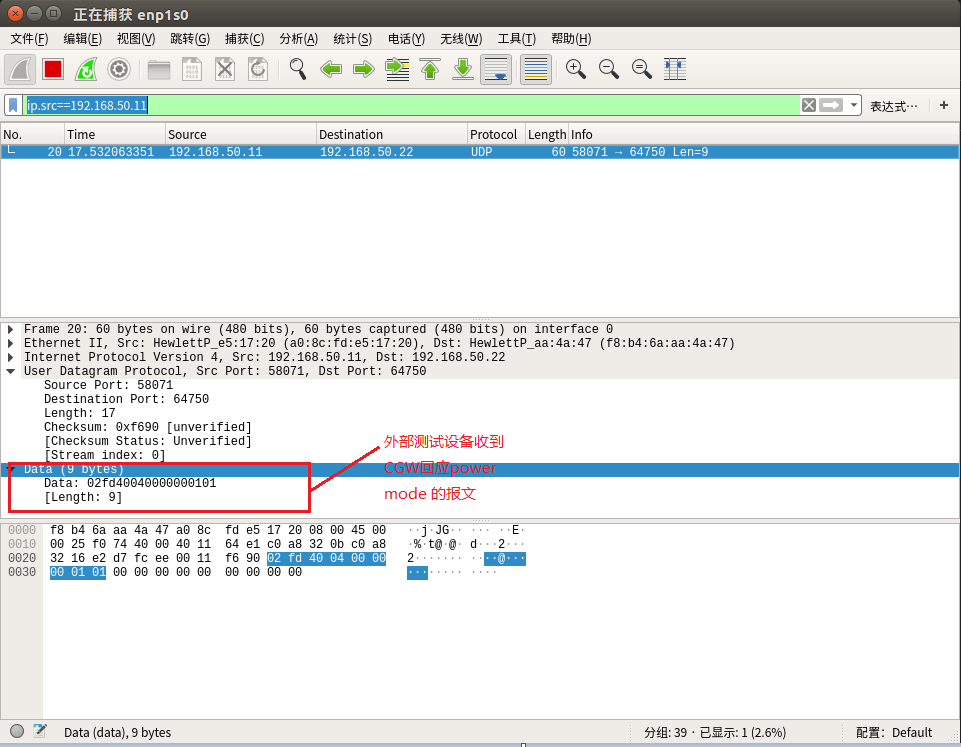
字节段02代表：Protocol version

字节段fd代表：Inverse protocol version

字节段4003代表：Diagnostic power mode information request

字节段00000000代表：Payload length(DoIP协议头之后的数据大小)

edgegateway回应消息给外部测试设备的UDP报文如下图所示(外部测试设备收到的UDP报文)：



字节段02代表：Protocol version

字节段fd代表：Inverse protocol version

字节段4004代表：Diagnostic power mode information response

字节段00000001代表：Payload length(DoIP协议头之后的数据大小)

之后的字节代表powermode实际的数据：01代表ready 00代表notready。

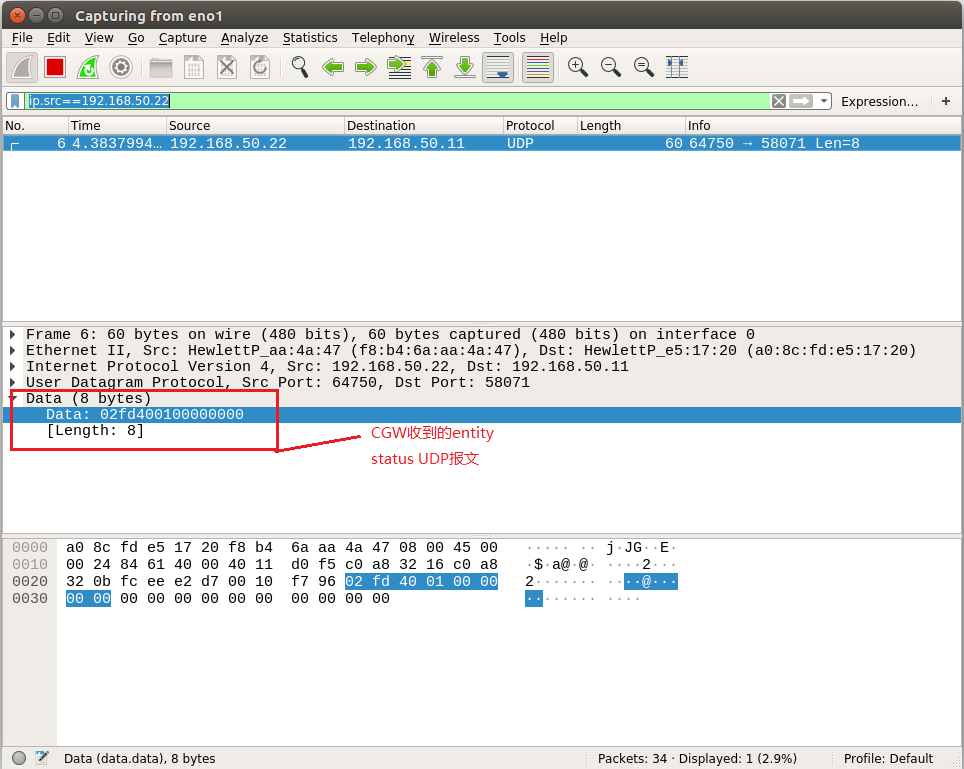
### DoIPEntitystatus报文分析

Entitystatus用于查询DoIP实体自身的一些状态，比如能够处理的最大数据大小，socket最大连接数，当前socket连接数，实体类型如gateway还是node。

发送entitystatus请求方：外部测试设备

接收并回应方：edgegateway（如MPU）

edgegateway收到的entitystatusUDP报文如下图所示：



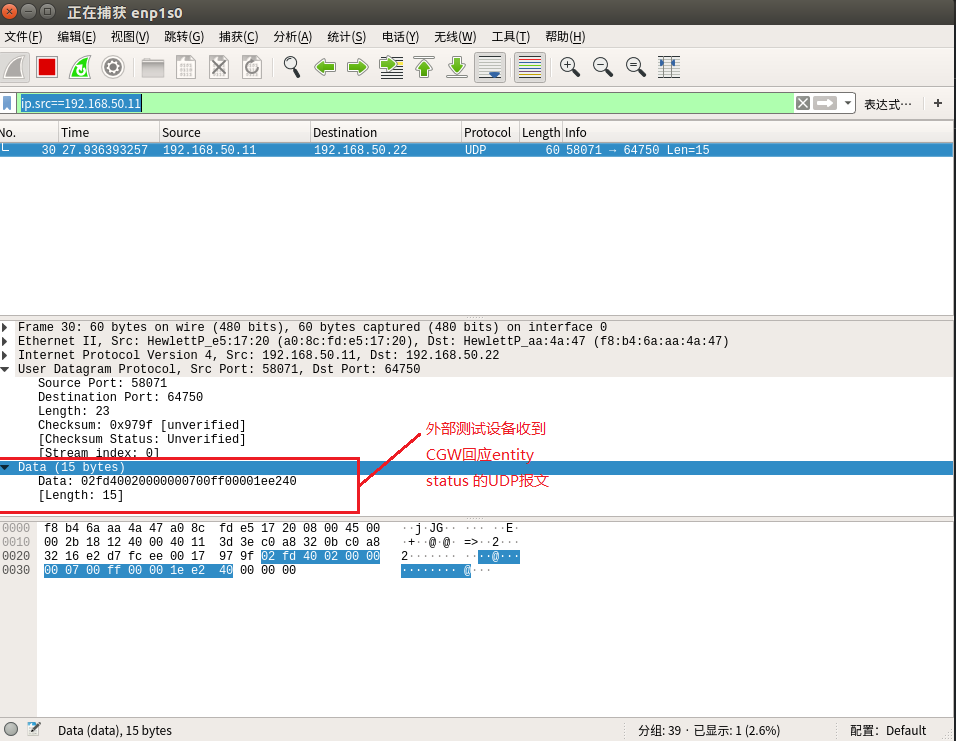
字节段02代表：Protocol version

字节段fd代表：Inverse protocol version

字节段4001代表：DoIP entity status requeste

字节段00000000代表：Payload length(DoIP协议头之后的数据大小)

edgegateway回应消息给外部测试设备的UDP报文如下图所示(外部测试设备收到的UDP报文)：



字节段02代表：Protocol version

字节段fd代表：Inverse protocol version

字节段4002代表：DoIP entity status response

字节段00000007代表：Payload length(DoIP协议头之后的数据大小)

之后的字节代表entitystatus实际的数据：00(Node type)，FF(Max. concurrent TCP\_DATA sockets)，0000(Currently open TCP\_DATA sockets)，1EE240(Max. data size)

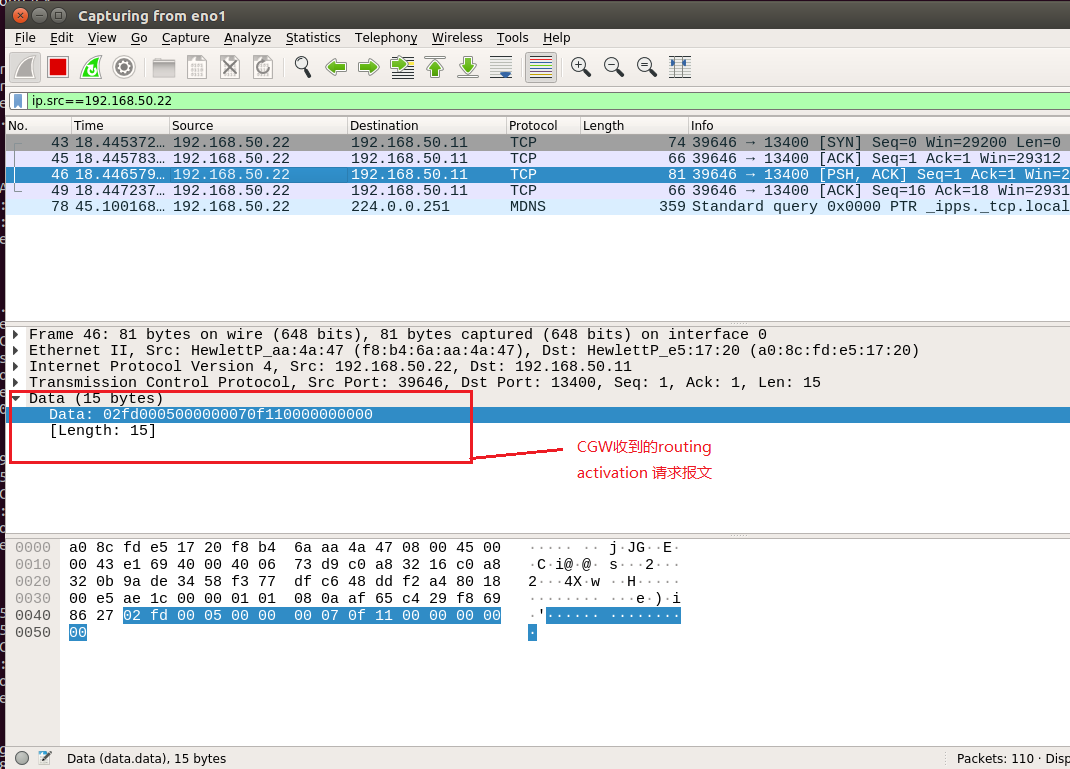
### DoIProuting activation报文分析

routing activation用于与DoIP实体建立TCP连接，路由的激活是诊断请求能够发送的前提条件。

发送routing activation请求方：外部测试设备

接收并回应方：edgegateway

edgegateway收到的routing activation TCP报文如下图所示：



CGW回应到外部测试设备的UDP报文如下图所示(外部测试设备收到的UDP报文)：

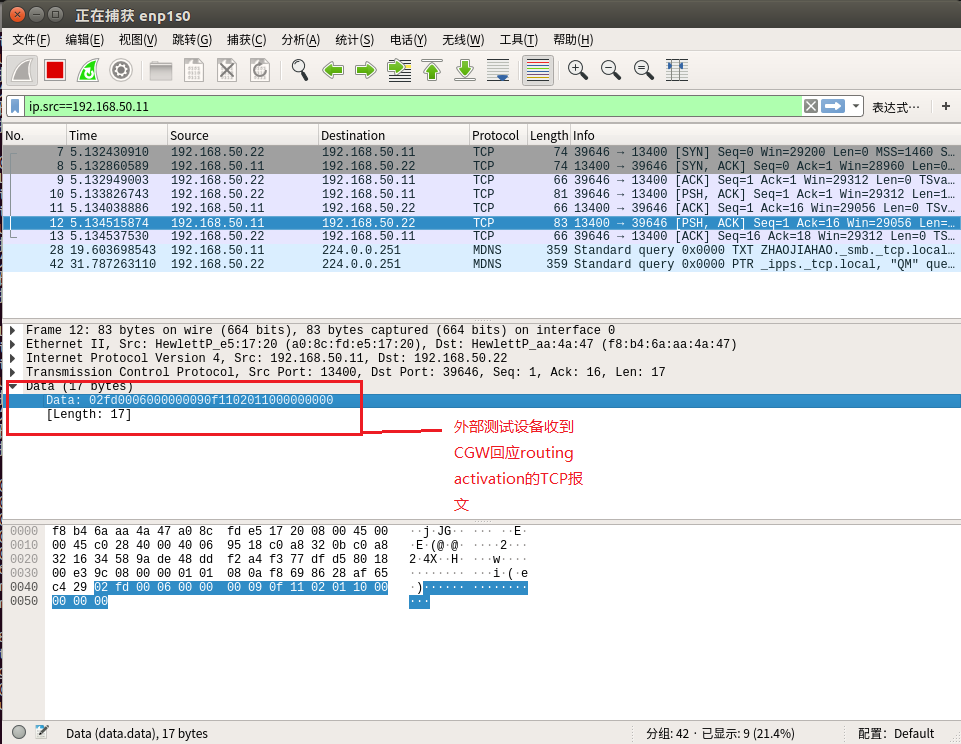
字节段02代表：Protocol version

字节段fd代表：Inverse protocol version

字节段0005代表：Routing activation requeste

字节段00000007代表：Payload length(DoIP协议头之后的数据大小)

之后的字节代表routing activation request实际的数据：0f11(Source address)，00(Activation type)，00(Reserved by this part of ISO 13400)，00000000(Reserved for OEMspecific use)



字节段02代表：Protocol version

字节段fd代表：Inverse protocol version

字节段0006代表：Routing activation response

字节段00000009代表：Payload length(DoIP协议头之后的数据大小)

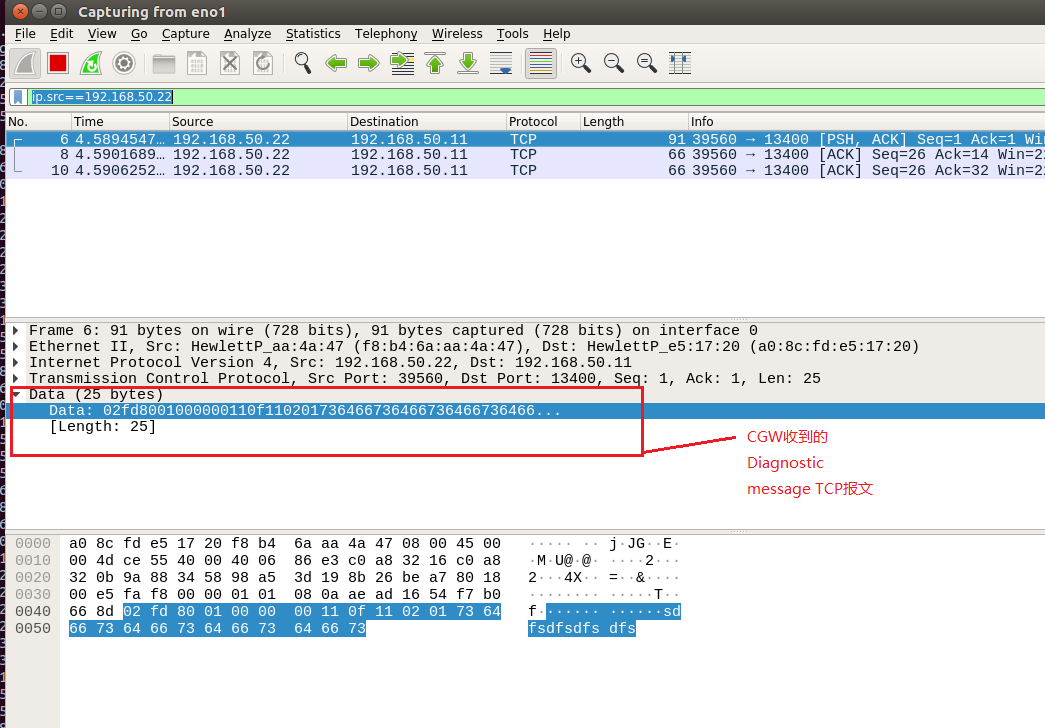
之后的字节代表routing activation response实际的数据：0f11(Logical address of external test equipment)，0201(Logical address of DoIP entity)，00(Routing activation response code)，00(Reserved by this part of ISO 13400)，00000000(Reserved for OEMspecific use)

### DoIPDiagnostic message报文分析

发送Diagnostic message请求方：外部测试设备

接收并回应方：edgegateway

edgegateway收到的Diagnostic message TCP报文如下图所示：



字节段02代表：Protocol version

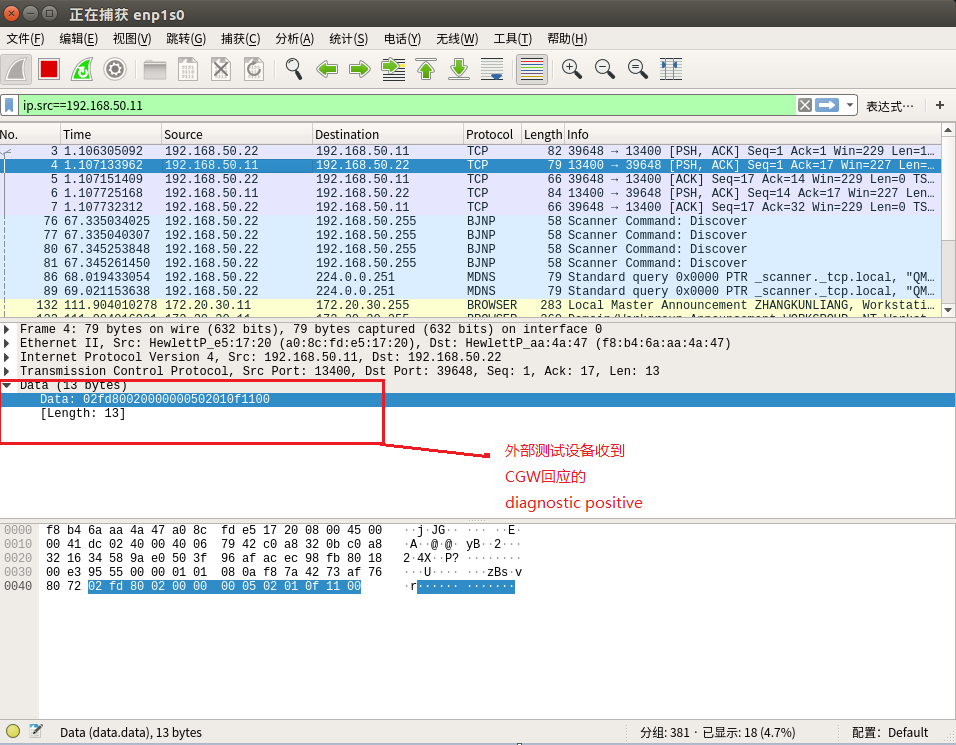
字节段fd代表：Inverse protocol version

字节段8001代表：Diagnostic message

字节段00000011代表：Payload length(DoIP协议头之后的数据大小)

之后的字节代表Diagnostic message实际的数据：诊断数据。

edgegateway回应消息给外部测试设备的UDP报文如下图所示(外部测试设备收到的UDP报文)：



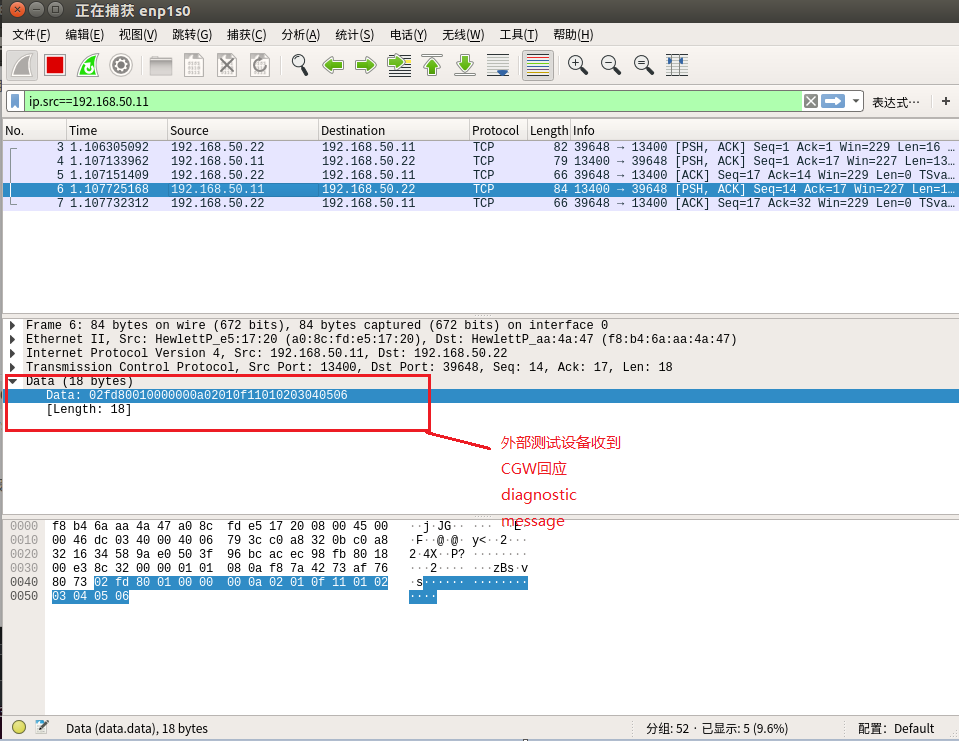
字节段02代表：Protocol version

字节段fd代表：Inverse protocol version

字节段8002代表：Diagnostic message positive acknowledgement

字节段00000005代表：Payload length(DoIP协议头之后的数据大小)

之后的字节代表Diagnostic message实际的数据：0201(Source address)，0f11(Target address)，00(ACK code)



字节段02代表：Protocol version

字节段fd代表：Inverse protocol version

字节段8001代表：Diagnostic message

字节段0000000a代表：Payload length(DoIP协议头之后的数据大小)

之后的字节代表Diagnostic message实际的数据：0201(Source address)，0f11(Target address)，010203040506(User data)

# 异常系处理

DoIP协议栈能够对异常系情况进行响应处理，使程序能够正常安全的运行下去。

## ECU异常

Case描述：在DoIP诊断程序发送诊断数据过程中，被诊断的ECU异常掉电，而ECU能够在6秒内重启。

程序处理：MPU发送诊断数据后，MPU端会启动A\_DoIP\_Diagnostic\_Message（2s）计时器，在计时器时间内没有收到ECU端的diagnostic\_massage\_ACK()报文，MPU端会进行第一次retry处理，再次发送诊断数据，retry处理最多两次；

ECU端在接收数据前重启成功，给MPU收到TCP重置报文，MPU重新进行路由激活并发送诊断数据。ECU成功接收完整的诊断数据后，给MPU发送diagnostic\_massage\_ACK()报文，MPU端的DoIP协议栈服务端会向上层UDSClient回应一个ACK（code：0x00）报文。

ECU异常能够在6秒内重启

UDS Client

DoIP

DoIP

UDS Server

request

routing activation request

routing activation request

diagnostic message

routing activation response

indication

routing activation response

tcp reset

diagnostic message PACK

req confirm(PACK)

diagnostic message

indication

底层TCP重置

retry start

ECU复位，6秒内成功启动

Case描述：在DoIP诊断程序发送诊断数据过程中，被诊断的ECU异常掉电，被诊断的ECU没有在6秒内重启完成。

程序处理：MPU发送诊断数据后，MPU端会启动A\_DoIP\_Diagnostic\_Message（2s）计时器，在计时器时间内没有收到ECU端的diagnostic\_massage\_ACK()报文，MPU会进行retry处理，再次发送诊断数据，retry处理最多两次。每次retry均会向上次UDS client发送NACK2（code：0xA0）。

ECU处于异常断电情况无法响应诊断数据，MPU在两次retry处理中都无法接收到ECU端的diagnostic\_massage\_ACK()报文，此时MPU端的DOIP协议栈服务端会向上层的UDS Client回应NACK（code：0x06）报文。

UDS Client

DoIP

DoIP

UDS Server

request

routing activation request

diagnostic message

diagnostic message

routing activation response

indication

req confirm(NACK)

diagnostic message

timeout

ECU离线，失去响应超过6秒

timeout

timeout

req confirm(NACK2)

req confirm(NACK2)

ECU异常无法在6秒内重启

## 协议解析异常

Case描述：Generic DoIP header进行解析时的异常处理，参考ISO13400-2 图例7 DoIPgenericheaderhandler。

程序处理：根据ISO13400-2 Table 14的动作进行实装。

Case描述：路由激活请求解析时的异常处理，参考ISO13400-2 图例9 DoIProutingactivationhandler。

程序处理：根据ISO13400-2 Table 25的动作进行实装。

Case描述：诊断消息请求解析时的异常处理，参考ISO13400-2 图例10 DoIPdiagnosticmessagehandler。

程序处理：根据ISO13400-2 Table31的动作进行实装。

## 其他异常

Case描述：在DoIP诊断程序进行路由激活过程中，因异常情况导致TCPconnect超时，TCP无法建立连接。

程序处理：TCPconnect超时后，MPU会再次发送TCP connect请求，TCP connect请求最多发送两次；若两次的TCP connect请求都超时后，MPU端的DoIP协议栈服务端会向上层UDS Client回应NACK（0x06）报文；若能够在两次TCP connect请求中建立TCP连接，则程序正常进行。