六叶树手把收教你学

SOCKETCAN

类别	内容
关键词	六叶树 SOCKET CAN 教程
摘要	介绍 SOCKET CAN 工具使用教程。

文档记录

版本	日期	说明
V1.00	2024.06	创建文档

目录

六叶树手把收教你学 1
SOCKETCAN 1
1.简介 3
2.硬件准备 3
3.can-utils 工具 3
3.1 安装方式一：命令安装 4
3.2 安装方式二：源码安装 5
4.常用命令 9
5.代码接口调用 9
5.1 C++ 10
6.免责说明 13
7.联系我们 13

1.简介

根据 维基百科 的介绍，SocketCAN 是一个开源的 CAN 驱动以及网络堆栈。在 linux 中，传统的 CAN 驱动是基于字符设备(character device)模型的。一个典型的设备驱动实现，只允许一个进程访问一个设备，其他进程的访问会被阻塞。而且不同设备之间的驱动往往略有不同，这也给移植带来了不便。而 SocketCAN 使用了网络设备模型，允许多个应用同时访问同一个 CAN 设备，而一个应用也可以同时访问多个 CAN 总线。

2.硬件准备

必须有 SOCKETCAN 卡，并且已安装好驱动。是否安装好设备驱动，可以通过 ifconfig -a 命令查询，如果有 CAN0、CAN1 这样的设备说明设备驱动已安装好。六叶树的大部分产品都已支持 SOCKETCAN，不太确定的可以咨询六叶树官方客服，六叶树官网:www.liuyeshu.cn。

```
orangepi@orangepi4-lts:~$ ifconfig -a
can0: flags=128<NOARP>  mtu 16
        unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00  txqueuelen 10  (UNSPEC)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

3.can-utils 工具

CAN-utils 是由 Linux CAN 社区维护的一个项目，旨在提供一套全面的工具，帮助开发者和工程师有效地与 CAN 设备交互。项目中包含诸如 candump, canfilter, cantest 等实用程序，覆盖了从数据帧捕获到报文发送、过滤等多种功能。

命令行工具：

candump：实时显示接收到的 CAN 消息，类似于网络嗅探器中的 tcpdump。

canfilter：过滤并显示满足特定条件的 CAN 消息，可用于数据分析或故障排查。

cantest：用于测试 CAN 接口，可以发送预定义的数据包，并验证接收是否正确。

cansend：允许用户直接发送自定义的 CAN 消息到总线上。

支持多种 CAN 接口：

CAN-utils 兼容各种硬件接口，包括 socketCAN（Linux 内核自带的 CAN 驱动）和其他第三方驱动，确保在不同平台上都能顺利工作。

兼容性与扩展性

由于基于标准的 socketCAN 框架，CAN-utils 与 Linux 发行版的兼容性很好，同时也为其他操作系统上的移植奠定了基础。此外，项目结构清晰，易于理解和扩展，开发者可以根据需求添加新的工具或修改现有功能。

应用场景：

硬件测试：在开发 CAN 设备时，可以通过 cantest 和 cansend 验证硬件和软件的正确性。

系统调试：使用 candump 和 canfilter 可以深入理解 CAN 网络的行为，找出潜在的问题。

数据分析：收集和过滤 CAN 消息，用于性能优化或故障后分析。

自动化脚本：结合 shell 脚本或更高级的语言，可以实现自动化测试和监控任务。

特点：

轻量级：作为纯命令行工具，CAN-utils 无需图形界面，运行效率高且资源占用少。

易用性：每个工具都有明确的参数说明，上手快速，适合初学者和专家使用。

灵活性：通过组合不同的工具和脚本，可以实现复杂的工作流程。

社区支持：作为 Linux CAN 的一部分，该项目有活跃的社区支持，更新及时，问题响应快。

3.1 安装方式一：命令安装

命令:

apt-get install can-utils

```
root@ubuntu:/home/zhumiao# apt-get install can-utils
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  can-utils
0 upgraded, 1 newly installed, 0 to remove and 255 not upgraded.
Need to get 0 B/117 kB of archives.
After this operation, 656 kB of additional disk space will be used.
Selecting previously unselected package can-utils.
(Reading database ... 198003 files and directories currently installed.)
Preparing to unpack .../can-utils_2018.02.0-1ubuntu1_amd64.deb ...
Unpacking can-utils (2018.02.0-1ubuntu1) ...
Setting up can-utils (2018.02.0-1ubuntu1) ...
Processing triggers for man-db (2.9.1-1) ...
```

检查安装结果:

命令:

candump --help

```
root@ubuntu:/home/zhumiao# candump -help
candump - dump CAN bus traffic.

Usage: candump [options] <CAN interface>+
  (use CTRL-C to terminate candump)

Options:                    可以输出这些帮助信息，说明安装成功！
         -t <type>   (timestamp: (a)bsolute/(d)elta/(z)ero/(A)bsolute w date)
         -H          (read hardware timestamps instead of system timestamps)
         -c          (increment color mode level)
         -i          (binary output - may exceed 80 chars/line)
         -a          (enable additional ASCII output)
         -S          (swap byte order in printed CAN data[] - marked with '`' )
         -s <level>  (silent mode - 0: off (default) 1: animation 2: silent)
         -b <can>    (bridge mode - send received frames to <can>)
         -B <can>    (bridge mode - like '-b' with disabled loopback)
         -u <usecs>  (delay bridge forwarding by <usecs> microseconds)
         -l          (log CAN-frames into file. Sets '-s 2' by default)
         -L          (use log file format on stdout)
         -n <count>  (terminate after receiption of <count> CAN frames)
         -r <size>   (set socket receive buffer to <size>)
         -D          (Don't exit if a "detected" can device goes down.
         -d          (monitor dropped CAN frames)
         -e          (dump CAN error frames in human-readable format)
         -x          (print extra message infos, rx/tx brs esi)
         -T <msecs>  (terminate after <msecs> without any reception)

Up to 16 CAN interfaces with optional filter sets can be specified
on the commandline in the form: <ifname>[,filter]*
```

3.2 安装方式二：源码安装

3.2.1 libsocketcan(依赖库)安装

1.源码下载（注意:源码下载地址可能会失效，网络搜索下载即可，通用）

wget https://public.pengutronix.de/software/libsocketcan/libsocketcan-0.0.11.tar.bz2

```
root@ubuntu:/home/zhumiao/tmp/tmp# wget https://public.pengutronix.de/software/libsocketcan/libsocketcan-0.0.11.tar.bz2
--2024-06-04 00:01:48--  https://public.pengutronix.de/software/libsocketcan/libsocketcan-0.0.11.tar.bz2
Resolving public.pengutronix.de (public.pengutronix.de)... 95.216.103.100, 2a01:4f9:2a:2c17:5054:ff:fe65:8c93
Connecting to public.pengutronix.de (public.pengutronix.de)|95.216.103.100|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 294663 (288K) [application/x-bzip2]
Saving to: 'libsocketcan-0.0.11.tar.bz2'

libsocketcan-0.0.11.tar.bz2       100%[==========================================================>] 287.76K   352KB/s    in 0.8s

2024-06-04 00:01:49 (352 KB/s) - 'libsocketcan-0.0.11.tar.bz2' saved [294663/294663]
```

2.源码解压

tar -xvf libsocketcan-0.0.11.tar.bz2

3.编译并安装

cd libsocketcan-0.0.11

./configure

```
checking for an ANSI C-conforming const... yes
checking for inline... inline
checking for size_t... yes
checking whether time.h and sys/time.h may both be included... yes
checking for working memcmp... yes
checking whether lstat correctly handles trailing slash... yes
checking whether stat accepts an empty string... no
checking for utime.h... (cached) yes
checking whether utime accepts a null argument... yes
checking for gethostbyaddr... yes
checking for gethostbyname... yes
checking for gethostname... yes
checking for gettimeofday... yes
checking for memset... yes
checking for mkdir... yes
checking for socket... yes
checking for utime... yes
checking whether to enable debugging... no
checking whether to enable error logging... yes
checking that generated files are newer than configure... done
configure: creating ./config.status
config.status: creating GNUmakefile
config.status: creating config/libsocketcan.pc
config.status: creating config/GNUmakefile
config.status: creating include/GNUmakefile
config.status: creating src/GNUmakefile
config.status: creating include/libsocketcan_config.h
config.status: executing libtool commands
config.status: executing depfiles commands
root@ubuntu:/home/zhumiao/tmp/tmp/libsocketcan-0.0.11#
```

make

```
root@ubuntu:/home/zhumiao/tmp/tmp/libsocketcan-0.0.11# make
Making all in include
make[1]: Entering directory '/home/zhumiao/tmp/tmp/libsocketcan-0.0.11/include'
make  all-am
make[2]: Entering directory '/home/zhumiao/tmp/tmp/libsocketcan-0.0.11/include'
make[2]: Leaving directory '/home/zhumiao/tmp/tmp/libsocketcan-0.0.11/include'
make[1]: Leaving directory '/home/zhumiao/tmp/tmp/libsocketcan-0.0.11/include'
Making all in config
make[1]: Entering directory '/home/zhumiao/tmp/tmp/libsocketcan-0.0.11/config'
make[1]: Nothing to be done for 'all'.
make[1]: Leaving directory '/home/zhumiao/tmp/tmp/libsocketcan-0.0.11/config'
Making all in src
make[1]: Entering directory '/home/zhumiao/tmp/tmp/libsocketcan-0.0.11/src'
  CC       libsocketcan.lo
  CCLD     libsocketcan.la
ar: `u' modifier ignored since `D' is the default (see `U')
make[1]: Leaving directory '/home/zhumiao/tmp/tmp/libsocketcan-0.0.11/src'
make[1]: Entering directory '/home/zhumiao/tmp/tmp/libsocketcan-0.0.11'
make[1]: Nothing to be done for 'all-am'.
make[1]: Leaving directory '/home/zhumiao/tmp/tmp/libsocketcan-0.0.11'
root@ubuntu:/home/zhumiao/tmp/tmp/libsocketcan-0.0.11#
```

make install

```
root@ubuntu:/home/zhumiao/tmp/tmp/libsocketcan-0.0.11# make install
Making install in include
make[1]: Entering directory '/home/zhumiao/tmp/tmp/libsocketcan-0.0.11/include'
make[2]: Entering directory '/home/zhumiao/tmp/tmp/libsocketcan-0.0.11/include'
make[2]: Nothing to be done for 'install-exec-am'.
 /usr/bin/mkdir -p '/usr/local/include'
 /usr/bin/install -c -m 644  libsocketcan.h can_netlink.h '/usr/local/include/.'
make[2]: Leaving directory '/home/zhumiao/tmp/tmp/libsocketcan-0.0.11/include'
make[1]: Leaving directory '/home/zhumiao/tmp/tmp/libsocketcan-0.0.11/include'
Making install in config
make[1]: Entering directory '/home/zhumiao/tmp/tmp/libsocketcan-0.0.11/config'
make[2]: Entering directory '/home/zhumiao/tmp/tmp/libsocketcan-0.0.11/config'
make[2]: Nothing to be done for 'install-exec-am'.
 /usr/bin/mkdir -p '/usr/local/lib/pkgconfig'
 /usr/bin/install -c -m 644 libsocketcan.pc '/usr/local/lib/pkgconfig'
make[2]: Leaving directory '/home/zhumiao/tmp/tmp/libsocketcan-0.0.11/config'
make[1]: Leaving directory '/home/zhumiao/tmp/tmp/libsocketcan-0.0.11/config'
Making install in src
make[1]: Entering directory '/home/zhumiao/tmp/tmp/libsocketcan-0.0.11/src'
make[2]: Entering directory '/home/zhumiao/tmp/tmp/libsocketcan-0.0.11/src'
```

3.2.2 can-utils 安装

1.源码下载（注意:源码下载地址可能会失效，网络搜索下载即可，通用）
wget https://public.pengutronix.de/software/socket-can/canutils/v4.0/canutils-4.0.6.tar.bz2

```
root@ubuntu:/home/zhumiao/tmp/tmp# wget https://public.pengutronix.de/software/socket-can/canutils/v4.0/canutils-4.0.6.tar.bz2
--2024-06-04 00:09:24--  https://public.pengutronix.de/software/socket-can/canutils/v4.0/canutils-4.0.6.tar.bz2
Resolving public.pengutronix.de (public.pengutronix.de)... 95.216.103.100, 2a01:4f9:2a:2c17:5054:ff:fe65:8c93
Connecting to public.pengutronix.de (public.pengutronix.de)|95.216.103.100|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 247783 (242K) [application/x-bzip2]
Saving to: 'canutils-4.0.6.tar.bz2'

canutils-4.0.6.tar.bz2            100%[=============================================================>] 241.98K   325KB/s    in 0.7s

2024-06-04 00:09:26 (325 KB/s) - 'canutils-4.0.6.tar.bz2' saved [247783/247783]

root@ubuntu:/home/zhumiao/tmp/tmp#
```

2.源码解压
tar -xvf canutils-4.0.6.tar.bz2
3.编译并安装
cd canutils-4.0.6/
./configure

```
root@ubuntu:/home/zhumiao/tmp/tmp/canutils-4.0.6# ./configure
checking build system type... x86_64-unknown-linux-gnu
checking host system type... x86_64-unknown-linux-gnu
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C89... none needed
checking for a sed that does not truncate output... /usr/bin/sed
checking for grep that handles long lines and -e... /usr/bin/grep
checking for egrep... /usr/bin/grep -E
checking for fgrep... /usr/bin/grep -F
checking for ld used by gcc... /usr/bin/ld
checking if the linker (/usr/bin/ld) is GNU ld... yes
checking for BSD- or MS-compatible name lister (nm)... /usr/bin/nm -B
checking the name lister (/usr/bin/nm -B) interface... BSD nm
checking whether ln -s works... yes
```

make

```
root@ubuntu:/home/zhumiao/tmp/tmp/canutils-4.0.6# make
Making all in include
make[1]: Entering directory '/home/zhumiao/tmp/tmp/canutils-4.0.6/include'
make  all-am
make[2]: Entering directory '/home/zhumiao/tmp/tmp/canutils-4.0.6/include'
make[2]: Leaving directory '/home/zhumiao/tmp/tmp/canutils-4.0.6/include'
make[1]: Leaving directory '/home/zhumiao/tmp/tmp/canutils-4.0.6/include'
Making all in config
make[1]: Entering directory '/home/zhumiao/tmp/tmp/canutils-4.0.6/config'
make[1]: Nothing to be done for 'all'.
make[1]: Leaving directory '/home/zhumiao/tmp/tmp/canutils-4.0.6/config'
Making all in src
make[1]: Entering directory '/home/zhumiao/tmp/tmp/canutils-4.0.6/src'
gcc -DHAVE_CONFIG_H -I. -I../include  -I../include -I../include -DPF_CAN=29 -DAF_CAN=PF_CAN    -Wall -g -O2
/candump.Tpo -c -o candump.o candump.c
```

make install

```
root@ubuntu:/home/zhumiao/tmp/tmp/canutils-4.0.6# make install
Making install in include
make[1]: Entering directory '/home/zhumiao/tmp/tmp/canutils-4.0.6/include'
make[2]: Entering directory '/home/zhumiao/tmp/tmp/canutils-4.0.6/include'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/home/zhumiao/tmp/tmp/canutils-4.0.6/include'
make[1]: Leaving directory '/home/zhumiao/tmp/tmp/canutils-4.0.6/include'
Making install in config
make[1]: Entering directory '/home/zhumiao/tmp/tmp/canutils-4.0.6/config'
make[2]: Entering directory '/home/zhumiao/tmp/tmp/canutils-4.0.6/config'
make[2]: Nothing to be done for 'install-exec-am'.
test -z "/usr/local/lib/pkgconfig" || /usr/bin/mkdir -p "/usr/local/lib/pkgconfig"
 /usr/bin/install -c -m 644 canutils.pc '/usr/local/lib/pkgconfig'
make[2]: Leaving directory '/home/zhumiao/tmp/tmp/canutils-4.0.6/config'
make[1]: Leaving directory '/home/zhumiao/tmp/tmp/canutils-4.0.6/config'
Making install in src
make[1]: Entering directory '/home/zhumiao/tmp/tmp/canutils-4.0.6/src'
make[2]: Entering directory '/home/zhumiao/tmp/tmp/canutils-4.0.6/src'
test -z "/usr/local/bin" || /usr/bin/mkdir -p "/usr/local/bin"
  /bin/bash ../libtool   --mode=install /usr/bin/install -c candump cansend canecho cansequence '/usr/local/bin'
libtool: install: /usr/bin/install -c candump /usr/local/bin/candump
libtool: install: /usr/bin/install -c cansend /usr/local/bin/cansend
libtool: install: /usr/bin/install -c canecho /usr/local/bin/canecho
libtool: install: /usr/bin/install -c cansequence /usr/local/bin/cansequence
```

3.2.2 can-utils 使用

注意事项:使用 can-utils 工具前，必须通过”常用命令”章节的指令配置好波特率并开启，配置顺序：先用”波特率设置命令”设置波特率，再用”开启命令”开启 CAN。否则会报错，系统重启后需要重新配置，建议这些指令的执行做在系统的启动脚本里。

1.数据发送

//ID 3 位代表标准帧 8 位代表扩展帧 不足的补零

cansend can0 023#1122334455667788

//扩展数据帧

cansend can0 00000123#1122334455667788

//远程帧

cansend can0 00000123#R

2.数据接收

//接收数据并打印在窗口

candump can0

3.数据发送测试

可通过 cangen 命令自动生成发送数据

cangen can0 -g 1 -i -x

1:代表 1ms 发送 1 帧。

可能出现的问题:

write: No buffer space available

解决方案:
ifconfig can0 txqueuelen 1000

4.常用命令

当设备上有多个 CAN 通道时，系统会自动生成 CAN0、CAN1 这些实例，这里以 CAN0 为例。

4.1 开启 can 设备

ip link set up can0

4.2 关闭 can 设备

ip link set down can0

4.3 设置 can 设备波特率

ip link set can0 type can bitrate 500000

提示:波特率单位:Hz,这里的 500000，代表 500K。

4.4 显示 can 设备详细信息

ip -details link show can0

```
lys@ubuntu:~$ ip -details link show can0
3: can0: <NOARP,ECHO> mtu 16 qdisc noop state DOWN mode DEFAULT group default qlen 10
    link/can  promiscuity 0
    can state STOPPED restart-ms 0
          kvaser_usb: tseg1 1..16 tseg2 1..8 sjw 1..4 brp 1..64 brp-inc 1
          clock 24000000numtxqueues 1 numrxqueues 1 gso_max_size 65536 gso_max_segs 65535
lys@ubuntu:~$
```

ip -details link show can0

5.代码接口调用

应用代码可以使用 SocketCAN 套接字对设备进行控制，包括数据的接收和发送以及滤

波。这里要注意，代码里的 socketCAN 套接字是不能控制 CAN 卡的开启和关闭以及波特率的设置，这些都必须通过“常用命令”章节的命令配置好。具体调用逻辑流程，可以查看六叶树官方的 socketCAN 专栏:http://www.liuyeshu.cn/?page_id=1256,里面有代码用例。

5.1 C++

```
/**
*****************************************************************************
* @file    socketCanTest.cpp
* @author  lys
* @version V1.1.0
* @date    2023
* @brief
*
* @note
* SOCKET CAN收发使用用例
* 准备条件:
* 1.设备驱动已安装好,ifconfig -a 能够查看can0设备
* 2.已通过命令ip link配置好波特率并开启
    ip link set can0 type can bitrate 500000
    ip link set up can0
* 用法:
* write:发送can数据,使用结构体can_frame
* read:接收can数据,使用结构体can_frame
* 代码功能:
* 程序运行,初始化了socket,设置了过滤器功能(setsockopt)，只接收0x590的报文，发送了1帧0x610
* 报文，接收到0x590的报文接收线程会自动退出，整个程序退出。
* @endverbatim
*****************************************************************************
* 版权:长沙六叶树教育科技有限公司
* 官网:www.liuyeshu.cn
* 更多资料教程下载见官网
*****************************************************************************
修改日期     版本号    修改者    功能描述
2018.11.10 V1.1.0  LYS
-----------------------------------------------------------------------------
*****************************************************************************
*/
#include <iostream>
#include <string.h>
#include <unistd.h>
#include <net/if.h>
#include <sys/ioctl.h>
#include <sys/socket.h>
#include <linux/can.h>
#include <linux/can/raw.h>
#include <thread>
```

```
/**
* @输入参数: 无
* @输出参数: 无
* @返 回 值: 无
* @说    明:接收线程函数
* 接收到1帧数据自动退出
*/
void receiveThread(int socket)
{
    struct can_frame frame;

    std::cout << "Info: read start...\n";

    int nbytes = read(socket, &frame, sizeof(struct can_frame));

    if (nbytes < 0)
    {
        std::cout << "Error: can raw socket read.\n";
        return;
    }

    if (nbytes < sizeof(struct can_frame))
    {
        std::cout << "Error: read incomplete CAN frame.\n";
        return;
    }

    printf("Receive OK\n");

    // 打印接收到的报文数据
    printf("ID=0x%X DLC=%d Data: ", frame.can_id, frame.can_dlc);
    for (uint8_t i = 0; i < frame.can_dlc; ++i)
    {
        printf("0x%02X ", frame.data[i]);
    }
    printf("\n");
}
```

```
int main()
{
    int nbytes;
    struct sockaddr_can socketCan0Addr;
    struct ifreq ifr;
    struct can_frame frame = {0};

    int socketFd = socket(PF_CAN, SOCK_RAW, CAN_RAW);

    // 获取can0的接口索引
    strcpy(ifr.ifr_name, "can0");
    ioctl(socketFd, SIOCGIFINDEX, &ifr);

    socketCan0Addr.can_family = AF_CAN;
    //把can0的接口索引赋值给socketCan0Addr
    socketCan0Addr.can_ifindex = ifr.ifr_ifindex;

    // 把socketCan0Addr绑定到socket上
    bind(socketFd, (struct sockaddr *)&socketCan0Addr, sizeof(socketCan0Addr));

    /*设置过滤，只接收id是0x590的CAN报文,如果需要接收所有数据
    ,不执行CAN_RAW_FILTER操作即可*/
    struct can_filter rfilter;
    rfilter.can_id = 0x590;
    rfilter.can_mask = CAN_SFF_MASK;
    setsockopt(socketFd, SOL_CAN_RAW, CAN_RAW_FILTER, &rfilter, sizeof(rfilter));

    // 开启接收回复的线程
    std::thread t(receiveThread, socketFd);

    // 准备数据
    frame.can_id = 0x610;
    //扩展帧
    frame.can_id |= CAN_EFF_FLAG;
    frame.can_dlc = 8;
    frame.data[0] = 0x40;
    frame.data[1] = 0x01;
    frame.data[2] = 0x10;
    frame.data[3] = 0x00;
    frame.data[4] = 0x00;
    frame.data[5] = 0x00;
    frame.data[6] = 0x00;
    frame.data[7] = 0x00;

    // 发送数据
    nbytes = write(socketFd, &frame, sizeof(frame));
    if (nbytes != sizeof(frame))
    {
        std::cout << "Send Error\n";
    }

    //等待接收线程结束
    t.join();

    //程序退出
    return 0;
}
```

这里做了数据过滤，如果需要接收所有数据，不执行该步骤即可。

6.免责说明

本文档提供有关长沙六叶树教育科技有限公司产品的信息。本文档并未授予任何知识产权的许可，并未以明示或暗示，或以禁止发言或其它方式授予任何知识产权许可。除六叶树教育科技在其产品的销售条款和条件中声明的责任之外，六叶树概不承担任何其它责任。并且，六叶树电子产品的销售使用不作任何明示或暗示的担保，包括对产品的特定用途适用性、适销性或对任何专利权、版权或其它知识产权的侵权责任等，均不作担保。六叶树可能随时对产品规格及产品描述做出修改，不另行通知。

7.联系我们


电话:15211065817(业务合作咨询)


邮箱:798746621@qq.com(业务咨询+技术支持)


微信:18673379565(技术支持)


官网:www.liuyeshu.cn(资料下载)


网上商城:https://shop112408209.taobao.com(产品购买)

淘宝店铺搜索:“六叶树教育科技”