make-guile

make APP\_BIN\_NAME=demo\_tuya\_ipc build\_app

tuyaf8a020b5b372197e b8d98EOgv6SlKob2woPHA1bcsFofNvVk

tuyaf89bd85132613bc5 5bWNe84VxNjL1e6eAaBfOkiAlxpktVJn

这是两组开发用的uuid&authkey，你保存一下，以后长期要用的。成龙会教你怎么用

./build\_app.sh demo\_tuya\_ipc wifi

./tuya\_ipc\_demo\_tuya\_ipc -m 2 -r ../ -t AYMkIbcxz8tWR6

./tuya\_ipc\_demo\_tuya\_ipc -m 2 -r ../../demo\_tuya\_ipc/ -t AY0kMlwe2tnzI3

./tuya\_ipc\_demo\_tuya\_ipc -m 2 -r ../../demo\_tuya\_ipc -t AYAQW8IEbiWAMw

../../demo\_tuya\_ipc/

"p":"tuya.140616","s":"Tuya","t":"AYdWrK4X5Fv7PY"}

sudo script screen.log

exit

ens33

AYDKl6e3UEbaWn

AYUh3ikHKZHugP

AYUp4apfbtbBRs

点一：

/////////

如果ifconfig只显示了lo，

ifconfig -a 却正常显示ens33。那么可以按照如下的操作：

service network-manager stop

rm /var/lib/NetworkManager/NetworkManager.state

service network-manager start

即可恢复网络连接

//////

点二：

/////////

Dbg:Init Value.storage\_path /tmp/

Dbg:Init Value.upgrade\_file\_path /tmp/upgrade.file

Dbg:Init Value.sd\_base\_path /tmp/

Dbg:Init Value.product\_key dy5s9aaspstm5qqd

Dbg:Init Value.uuid tuyaf8a020b5b372197e

Dbg:Init Value.auth\_key b8d98EOgv6SlKob2woPHA1bcsFofNvVk

Dbg:Init Value.p2p\_id

Dbg:Init Value.dev\_sw\_version 1.2.3

Dbg:Init Value.max\_p2p\_user 5 //这里的5是代表在局域网下面有5个连接者吗

/////////

点三：coredump

#include <stdio.h>

int func(int \*p)

{

\*p = 0;

}

int main()

{

func(NULL);

return 0;

}

总结一下，需要定位进程挂在哪一行我们只需要4个操作，

ulimit -c unlimited

echo "/tmp/core-%e-%p" > /proc/sys/kernel/core\_pattern

gcc -o main -g a.c

gdb main /tmp/core-main-10815

在载入core的时候有个小技巧，如果你事先不知道这个core文件是由哪个程序产生的，你可以先随便找个代替一下，比如/usr/bin/w就是不错的选择。

比如我们采用这种方法载入上边产生的core，gdb会有类似的输出：

sagi@sagi-laptop:~$ gdb /usr/bin/w core

Core was generated by ./coredump'.

Program terminated with signal 11, Segmentation fault.

#0 0x080483a7 in ?? ()

(gdb)

可以看到GDB已经提示你了，这个core是由哪个程序产生的。

点4：

执行strip a.out，然后执行ls -l a.out看一下文件大小，用file命令来查看文件基本信息

的，用nm命令来列出一个目标文件中的各种符号。很明显，文件已经变小了

储存一部25帧每秒，90分钟，分辨率为1024\*768，24位真彩色的电影，若没有经过压缩，储存视频信息需要的储存空间大约为?

1024\*768\*24\*25\*60\*90个二进制位 bit = 296.63085GByte

点五：

strip命令用于脱掉文件的衣服， 文件会变小， 其中的符号信息会失去。 那这个strip有什么

用呢？ 很有用的！ 原来的a.out比较大， 可以执行。 在strip之后， 文件变小了， 也是可以执行， 这就节省了很多空间.

其实， strip不仅仅可以针对可执行文件， 还能针对目标文件和动态库等.

在实际的开发中， 经常需要对动态库.so进行strip操作， 减少占地空间。 而在调试的时候（比如用addr2line）， 就需要符号了。

因此， 通常的做法是： strip前的库用来调试， strip后的库用来实际发布， 他们两者有对应关系。 一旦发布的strip后的库出了问题，

就可以找对应的未strip的库来定位

六：git的操作命令

即可获取到版本号：git reflog

git log命令显示从最近到最远的显示日志

git log –pretty=oneline

版本回退操作

git reset --hard HEAD^

git reset --hard HEAD~100

理解工作区与暂存区的区别？

　　工作区：就是你在电脑上看到的目录，比如目录下testgit里的文件(.git隐藏目录版本库除外)。

或者以后需要再新建的目录文件等等都属于工作区范畴。

　　版本库(Repository)：工作区有一个隐藏目录.git,这个不属于工作区，这是版本库。其中版本库里面存了很多东西，其中最重要的就是stage(暂存区)，

还有Git为我们自动创建了第一个分支master,以及指向master的一个指针HEAD。

　　我们前面说过使用Git提交文件到版本库有两步：

　　第一步：是使用 git add 把文件添加进去，实际上就是把文件添加到暂存区。

　　第二步：使用git commit提交更改，实际上就是把暂存区的所有内容提交到当前分支上。