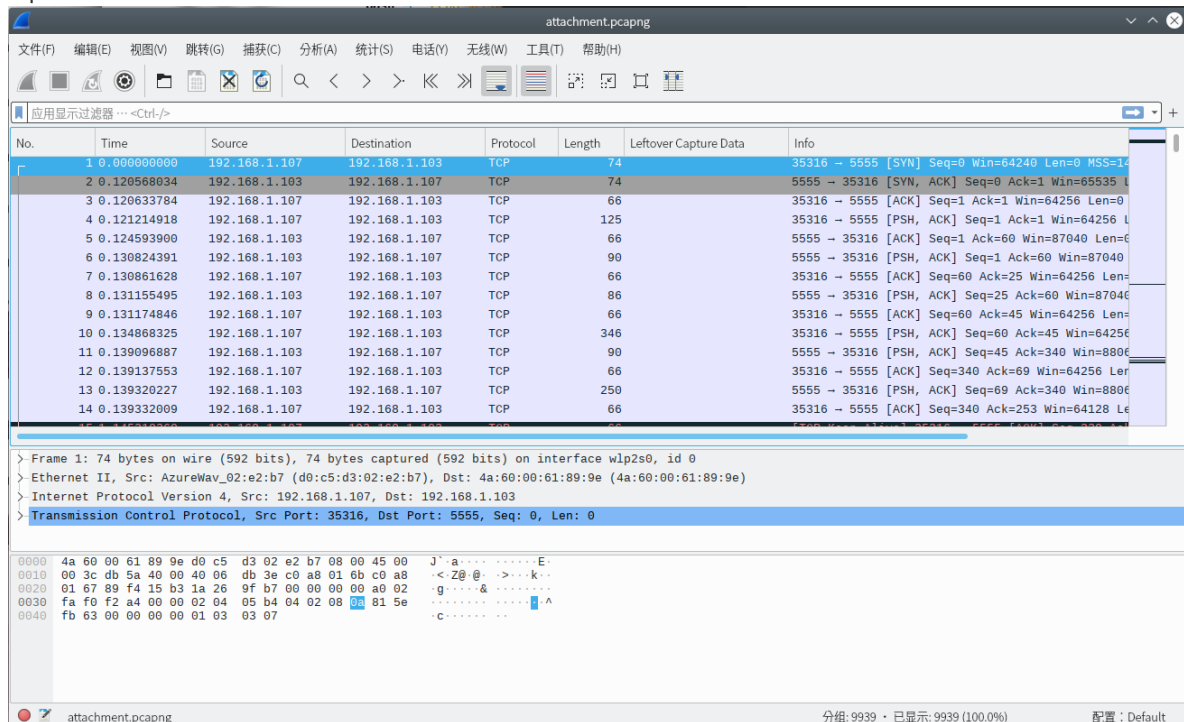


Open the attachment in Wireshark:



Port 5555, It's Network ADB, let's dig deeper.

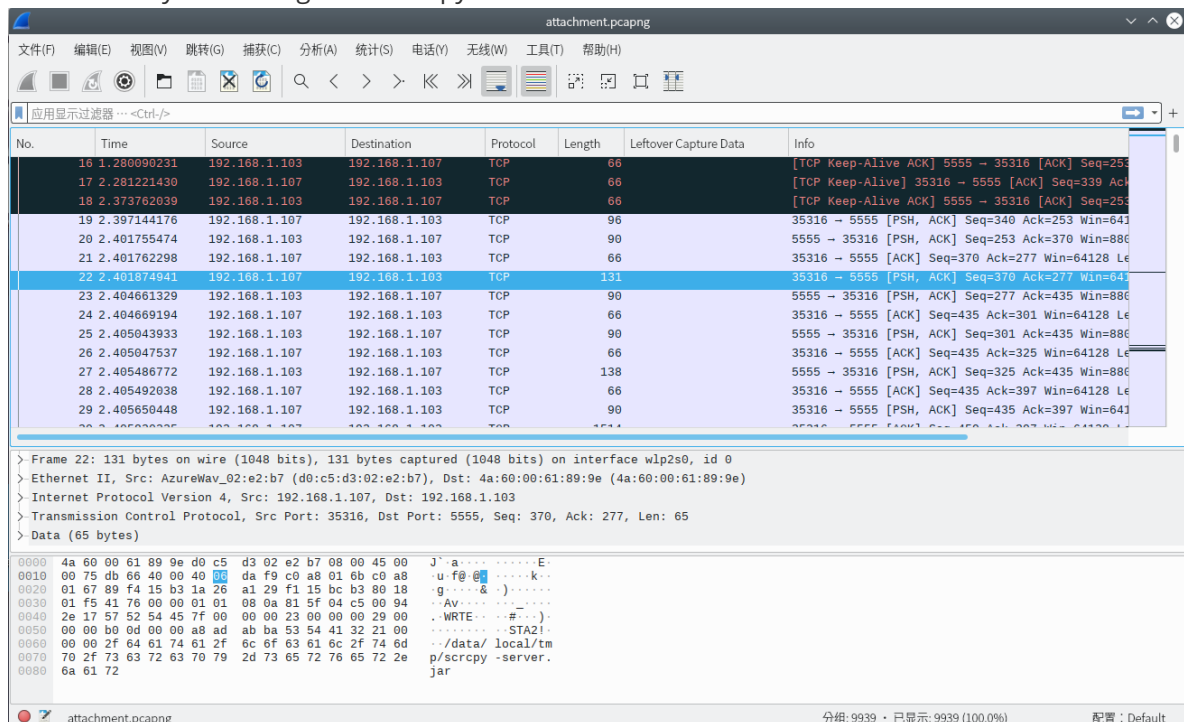
Generally we want to see the document of ADB protocol, which can be found here:

<https://github.com/cstyan/adbDocumentation>

You can know that **WRTE** means the packet is sent to the client.

you can also skip this if you directly found this packet, but it may be harder afterwards:

So it's actually something about scrpy.



Filter out packets sent from the compter:

what does these packets mean?

attachment.pcapng

文件(F) 编辑(E) 视图(V) 跳转(G) 捕获(C) 分析(A) 统计(S) 电话(Y) 无线(W) 工具(T) 帮助(H)

ip.dst_host == "192.168.1.103"

No.	Time	Source	Destination	Protocol	Length	Leftover Capture Data	Info
7026	32.576569095	192.168.1.107	192.168.1.103	TCP	90		35316 → 5555 [PSH, ACK] Seq=112488 Ack=3229397
7028	32.663094913	192.168.1.107	192.168.1.103	TCP	146		35316 → 5555 [PSH, ACK] Seq=112512 Ack=3229397
7031	32.713193179	192.168.1.107	192.168.1.103	TCP	66		35316 → 5555 [ACK] Seq=112592 Ack=3229421 Win=2
7032	32.713975823	192.168.1.107	192.168.1.103	TCP	118		35316 → 5555 [PSH, ACK] Seq=112592 Ack=3229421
7034	32.716826425	192.168.1.107	192.168.1.103	TCP	66		35316 → 5555 [ACK] Seq=112644 Ack=3229445 Win=2
7035	32.724109371	192.168.1.107	192.168.1.103	TCP	146		35316 → 5555 [PSH, ACK] Seq=112644 Ack=3229445
7037	32.728382798	192.168.1.107	192.168.1.103	TCP	66		35316 → 5555 [ACK] Seq=112724 Ack=3229469 Win=2
7038	32.728580224	192.168.1.107	192.168.1.103	TCP	118		35316 → 5555 [PSH, ACK] Seq=112724 Ack=3229469
7040	32.732669534	192.168.1.107	192.168.1.103	TCP	66		35316 → 5555 [ACK] Seq=112776 Ack=3229493 Win=2
7041	32.734340391	192.168.1.107	192.168.1.103	TCP	146		35316 → 5555 [PSH, ACK] Seq=112776 Ack=3229493
7043	32.736341726	192.168.1.107	192.168.1.103	TCP	66		35316 → 5555 [ACK] Seq=112856 Ack=3229517 Win=2
7046	32.736751532	192.168.1.107	192.168.1.103	TCP	66		35316 → 5555 [ACK] Seq=112856 Ack=3231017 Win=2
7047	32.736986290	192.168.1.107	192.168.1.103	TCP	90		35316 → 5555 [PSH, ACK] Seq=112856 Ack=3231017
7049	32.738095074	192.168.1.107	192.168.1.103	TCP	66		35316 → 5555 [ACK] Seq=112880 Ack=3231041 Win=2

> Frame 7035: 146 bytes on wire (1168 bits), 146 bytes captured (1168 bits) on interface wlp2s0, id 0
 > Ethernet II, Src: AzureWav_02:e2:b7 (d0:c5:d3:02:e2:b7), Dst: 4a:60:00:61:89:9e (4a:60:00:61:89:9e)
 > Internet Protocol Version 4, Src: 192.168.1.107, Dst: 192.168.1.103
 > Transmission Control Protocol, Src Port: 35316, Dst Port: 5555, Seq: 112644, Ack: 3229445, Len: 80
 > Data (80 bytes)

```

0000  4a 60 00 61 89 9e d0 c5 d3 02 e2 b7 08 00 45 00  J-a-...E-
0010  00 84 e7 36 40 00 40 06 cf 12 c0 a8 01 6b c0 a8  >@-...k-
0020  01 67 89 f4 15 b3 1a 28 57 bb f1 47 02 a3 80 18  g-...W-G-
0030  06 78 da 66 00 00 01 01 08 0a 81 5f 7b 37 00 94  x-F-...{7-
0040  39 ef 57 52 54 45 87 00 00 00 27 00 00 00 38 00  9WRTE-...8-
0050  00 00 c8 18 00 00 a8 ad ab ba 02 02 ff ff ff ff  .....<-B-
0060  ff ff ff ff 00 00 01 3c 00 00 02 fd 04 38 08 e8  .....<-B-
0070  ff ff 00 00 00 01 02 02 ff ff ff ff ff ff ff ff  .....<-B-
0080  00 00 01 3c 00 00 02 ff 04 38 08 e8 ff ff 00 00  .....<-B-
0090  00 01
  
```

Destination Host: Character string 分组: 9939 · 已显示: 4344 (43.7%) 配置: Default

Search the web and you can find this Github issue:

<https://github.com/Genymobile/scrcpy/issues/673>

says there's no documentation, but you can refer to the source code:

https://github.com/Genymobile/scrcpy/blob/6b3d9e3eab1d9ba4250300eccd04528dbbee9023a/app/tests/test_control_msg_serialize.c

```

static void test_serialize_inject_mouse_event(void) {
    struct control_msg msg = {
        .type = CONTROL_MSG_TYPE_INJECT_MOUSE_EVENT,
        .inject_mouse_event =
            {
                .action = AMOTION_EVENT_ACTION_DOWN,
                .buttons = AMOTION_EVENT_BUTTON_PRIMARY,
                .position =
                    {
                        .point =
                            {
                                .x = 260,
                                .y = 1026,
                            },
                        .screen_size =
                            {
                                .width = 1080,
                                .height = 1920,
                            },
                    },
            },
    };

    unsigned char buf[CONTROL_MSG_SERIALIZED_MAX_SIZE];
    int size = control_msg_serialize(&msg, buf);
    assert(size == 18);
    const unsigned char expected[] =
    { CONTROL_MSG_TYPE_INJECT_MOUSE_EVENT,
      0x00, // AKEY_EVENT_ACTION_DOWN
      0x00,
      0x00,
  
```

```

    0x00,
    0x01, // AMOTION_EVENT_BUTTON_PRIMARY
} 0x00,
    0x00, 0x01, 0x04, 0x00, 0x00, 0x04, 0x02, // 260 1026
    0x04, 0x38, 0x07, 0x80, // 1080 1920
};
assert(!memcmp(buf, expected, sizeof(expected)));

```

Then export the capture file as json and match patterns like above, extracting the points(X,Y):
(fish script, and 57:52:54:45 is the adb command WRTE you got from the documentation)

```

for i in (cat /home/leohearts/Desktop/tmp.json | jq .[]._source.layers.tcp | grep
57:52:54:45 | grep -o -E '00:00:.....:00:00:.....:04:38:08:e8:ff' | grep -o -E
'.....:' | grep -v "00:00" | grep -v '04:38' | grep -v '08:e8' | sed 's://g')
bash -c 'echo $((16#'$i'))' >> pixels
end

```

Then use Python to draw it back to an image:

```

from PIL import Image
def newImg():
    img = Image.new('RGB', (2000, 2000))
    while True:
        try:
            x = int(input())
            y = int(input())
            img.putpixel((x,y), (155,155,55))
        except:
            break
    img.save('sqr.png')
    return img

wallpaper = newImg()
wallpaper.show()

```

```
cat pixels | python3 image.py
```

Then thats the flag.

SC7Ff
better_
access_
with_sor
CPy3