MWR LABS

Security Advisory

Huawei Frame Buffer Driver Arbitrary Memory Write

18/07/2017

Software	MediaTek Frame Buffer Driver
Affected Versions	Huawei Y6 Pro Dual SIM (TIT-L01C576B115)
Author	Mateusz Fruba
Severity	High
Vendor	Huawei
Vendor Response	Fix Released

Description:

Huawei is a company that provides networking and telecommunications equipment.

The MediaTek frame buffer driver, as shipped with Huawei Y6 Pro, implements an IOCTL interface vulnerable to an arbitrary memory write due to insufficient input validation.

Impact:

Local processes running in the context of a system application, media server, or system server can leverage the frame buffer driver memory corruption to escalate their privileges to root or kernel.

Cause:

The MediaTek frame buffer driver fails to validate user-supplied data.

Solution:

This vulnerability was resolved by Huawei in version TIT-L01C576B119. More information can be found on the Huawei web page: http://www.huawei.com/en/psirt/security-advisories/huawei-sa-20170527-01-smartphone-en

Technical details

The MediaTek frame buffer driver implements the ‘mtkfb_ioctl‘ IOCTL handler which receives data passed from user space to the kernel. This driver is implemented in ‘/drivers/misc/mediatek/videox/mt6735/’.

When ‘mtkfb_ioctl’ is called, the ‘pbuf’ variable is user-controlled and copied from user space in ‘mtkfb_ioctl’ using ‘copy_from_user’. This can be seen below:

```
static int mtkfb_ioctl(struct fb_info *info, unsigned int cmd, unsigned long arg)
{
       void __user *argp = (void __user *)arg;
       DISP_STATUS ret = 0;
       int r = 0;

…
              switch (cmd)
              {
…
              case MTKFB_CAPTURE_FRAMEBUFFER:
                     {
                            unsigned long pbuf = 0;
…
                            else
                            {
                                   dprec_logger_start(DPREC_LOGGER_WDMA_DUMP, 0, 0);
                                   primary_display_capture_framebuffer_ovl(pbuf, eBGRA8888);
                                   dprec_logger_done(DPREC_LOGGER_WDMA_DUMP, 0, 0);
                            }
…
```

The ‘primary_display_capture_framebuffer_ovl’ function is then called with the ‘pbuf’ variable set to the value passed from user space.

The ‘primary_display_capture_framebuffer_ovl’ function makes use of ‘memset’ and casts the unsigned long value to a void pointer without any verification as follows:

```
int primary_display_capture_framebuffer_ovl(unsigned long pbuf, unsigned int format)
{
...
	if (pgc->state == DISP_SLEEPED)
	{
		memset((void*)pbuf, 0, buffer_size);
...
	}
	if (!primary_display_cmdq_enabled())
	{
		memset((void*)pbuf, 0, buffer_size);
...
	}

	if (_is_decouple_mode(pgc->session_mode) || _is_mirror_mode(pgc->session_mode))
	{
		primary_display_capture_framebuffer_decouple(pbuf, format);
		memset((void*)pbuf, 0, buffer_size);
...
			ret = m4u_alloc_mva(m4uClient, M4U_PORT_DISP_WDMA0, pbuf, NULL, 
buffer_size, M4U_PROT_READ | M4U_PROT_WRITE, 0, (unsigned int*)&mva);
...
			ret = m4u_cache_sync(m4uClient, M4U_PORT_DISP_WDMA0, pbuf, buffer_size, 
mva, M4U_CACHE_FLUSH_BY_RANGE);
```

This leads to control of the destination address for ‘memset’, thus allowing to zero out arbitrary memory.

Following this, we see that ‘pbuf’ is passed as an argument to ‘primary_display_capture_framebuffer_decouple’ and used without any validation while calculation the destination address for a write operation.

```
int primary_display_capture_framebuffer_decouple(unsigned long pbuf, unsigned int format)
{
	...
		for (i = 0; i<h_yres; i++) {
			for (j = 0; j<w_xres; j++) {
				memcpy((void*)tem_va, (void*)src_va, 4);
				*(unsigned long*)(pbuf + (i*w_xres + j) * 4) = 0xFF000000 | 
(tem_va[0] << 16) | (tem_va[1] << 8) | (tem_va[2]);
…
```

The lack of input validation in this function allows malicious attackers to provide values that can result in an arbitrary memory write.

The following crash is observed when this vulnerability is triggered:

```
2652.261167]<0> (0)[6016:MTKFB_CAPTURE_F][DISP]func|mtkfb_open
[ 2652.262092]<0> (0)[6016:MTKFB_CAPTURE_F][DISP]func|mtkfb_ioctl
[ 2652.262107]<0> (0)[6016:MTKFB_CAPTURE_F]mtkfb_ioctl, info=ffffffc078442800, cmd
nr=0x00000003, cmd size=0x00000008
[ 2652.262119]<0> (0)[6016:MTKFB_CAPTURE_F][DISP]primary capture: begin
[ 2652.262136]<0> (0)[6016:MTKFB_CAPTURE_F]Unable to handle kernel paging request at
virtual address 4142424244
[ 2652.262148]<0> (0)[6016:MTKFB_CAPTURE_F]pgd = ffffffc047571000
[ 2652.262157][4142424244] *pgd=0000000000000000
[ 2652.262170]<0> (0)[6016:MTKFB_CAPTURE_F][KERN Warning] ERROR/WARN forces debug_lock off!
[ 2652.262180]<0> (0)[6016:MTKFB_CAPTURE_F][KERN Warning] check backtrace:
…
[ 2659.832215]<0>-(0)[6016:MTKFB_CAPTURE_F]CPU: 0 PID: 6016 Comm: MTKFB_CAPTURE_F Tainted:
G        W    3.10.65 #1
[ 2659.832233]<0>-(0)[6016:MTKFB_CAPTURE_F]task: ffffffc04d30a000 ti: ffffffc0398c8000
task.ti: ffffffc0398c8000
[ 2659.832250]<0>-(0)[6016:MTKFB_CAPTURE_F]PC is at memset+0x1c/0x60
[ 2659.832268]<0>-(0)[6016:MTKFB_CAPTURE_F]LR is at
primary_display_capture_framebuffer_ovl+0x2b4/0x498
[ 2659.832281]<0>-(0)[6016:MTKFB_CAPTURE_F]pc : [<ffffffc0002e8f4c>] lr :
[<ffffffc000495840>] pstate: 20000145
[ 2659.832292]<0>-(0)[6016:MTKFB_CAPTURE_F]sp : ffffffc0398cba50
[ 2659.832301]x29: ffffffc0398cba50 x28: ffffffc0398c8000
[ 2659.832318]x27: ffffffc000dbf000 x26: ffffffc001077ef0
[ 2659.832334]x25: 0000000001002002 x24: 0000000000000500
[ 2659.832349]x23: 00000000000002d0 x22: 0000004142424244
[ 2659.832365]x21: ffffffc001077ea8 x20: 0000000000000000
[ 2659.832380]x19: ffffffc001077000 x18: 0000000000000001
[ 2659.832395]x17: 0000007f8737d350 x16: ffffffc00010dc0c
[ 2659.832410]x15: 0000000000000000 x14: 0000000000020000
[ 2659.832425]x13: 000000000001aa78 x12: 0000000000000001
[ 2659.832441]x11: 0000000000000058 x10: ffffffc000bb1bb8
[ 2659.832457]x9 : 00000000000042da x8 : 000000000000fc40
[ 2659.832472]x7 : 0000000000000048 x6 : 000000000000b9ae
[ 2659.832486]x5 : 000000000000000a x4 : 0000004142424244
[ 2659.832500]x3 : 0000000000000001 x2 : 0000000000383ff8
[ 2659.832516]x1 : 0000000000000000 x0 : 0000004142424244
```

As shown in the above crash log it was possible to force MediaTek frame buffer to put '0x4142424244' value as destination address for 'memset' function what caused kernel panic, as this user space address was currently not mapped within process address space.

Detailed Timeline

Date	Summary
2017-03-28	Issue reported to Huawei.
2017-06-05	Huawei confirmed this issue was fixed in version TIT-L01C576B119.