u-boot / **u-boot** Public

Code   Pull requests 53   Actions   Security   Insights

i2c: fix stack buffer overflow vulnerability in i2c md command

Browse files

```
When running "i2c md 0 0 80000100", the function do_i2c_md parses the
length into an unsigned int variable named length. The value is then
moved to a signed variable:

    int nbytes = length;
    #define DISP_LINE_LEN 16
    int linebytes = (nbytes > DISP_LINE_LEN) ? DISP_LINE_LEN : nbytes;
    ret = dm_i2c_read(dev, addr, linebuf, linebytes);

On systems where integers are 32 bits wide, 0x80000100 is a negative
value to "nbytes > DISP_LINE_LEN" is false and linebytes gets assigned
0x80000100 instead of 16.

The consequence is that the function which reads from the i2c device
(dm_i2c_read or i2c_read) is called with a 16-byte stack buffer to fill
but with a size parameter which is too large. In some cases, this could
trigger a crash. But with some i2c drivers, such as drivers/i2c/nx_i2c.c
(used with "nexell,s5pxx18-i2c" bus), the size is actually truncated to
a 16-bit integer. This is because function i2c_transfer expects an
unsigned short length. In such a case, an attacker who can control the
response of an i2c device can overwrite the return address of a function
and execute arbitrary code through Return-Oriented Programming.

Fix this issue by using unsigned integers types in do_i2c_md. While at
it, make also alen unsigned, as signed sizes can cause vulnerabilities
when people forgot to check that they can be negative.

Signed-off-by: Nicolas Iooss <nicolas.iooss+uboot@ledger.fr>
Reviewed-by: Heiko Schocher <hs@denx.de>
```

master

v2023.01-rc2 ... v2022.07-rc6

Nicolas Iooss authored and **trini** committed on Jun 28

1 parent b75fd37   commit 8f8c04bf1ebbd2f72f1643e7ad9617dafa6e5409

Showing **1 changed file** with **12 additions** and **12 deletions**.

Split   Unified

24 🟩🟩🟥🟥⬜ cmd/i2c.c

```
200  200     *
```

```diff
  * Returns the address length.
  */
- static uint get_alen(char *arg, int default_len)
+ static uint get_alen(char *arg, uint default_len)
  {
- 	int	j;
- 	int	alen;
+ 	uint	j;
+ 	uint	alen;

  	alen = default_len;
  	for (j = 0; j < 8; j++) {
  {
  	uint	chip;
  	uint	devaddr, length;
- 	int alen;
+ 	uint	alen;
  	u_char	*memaddr;
  	int ret;
  #if CONFIG_IS_ENABLED(DM_I2C)
  {
  	uint	chip;
  	uint	devaddr, length;
- 	int alen;
+ 	uint	alen;
  	u_char	*memaddr;
  	int ret;
  #if CONFIG_IS_ENABLED(DM_I2C)
  {
  	uint	chip;
  	uint	addr, length;
- 	int alen;
- 	int	j, nbytes, linebytes;
+ 	uint	alen;
+ 	uint	j, nbytes, linebytes;
  	int ret;
  #if CONFIG_IS_ENABLED(DM_I2C)
  	struct udevice *dev;
  {
  	uint	chip;
  	ulong	addr;
- 	int	alen;
+ 	uint	alen;
  	uchar	byte;
- 	int	count;
+ 	uint	count;
  	int ret;
  #if CONFIG_IS_ENABLED(DM_I2C)
  	struct udevice *dev;
```

```diff
 {
 	uint	chip;
 	ulong	addr;
-	int	alen;
-	int	count;
+	uint	alen;
+	uint	count;
 	uchar	byte;
 	ulong	crc;
 	ulong	err;
 			char *const argv[])
 {
 	uint	chip;
-	int alen;
+	uint	alen;
 	uint	addr;
 	uint	length;
 	u_char	bytes[16];
```

0 comments on commit `8f8c04b`

Please sign in to comment.