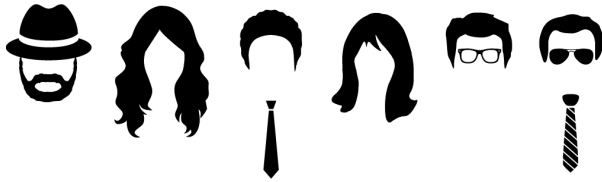


2 In Praise of Junk Hacking

by Pastor Manul Laphroaig
in polite dissent to Daily Dave.



Gather round y'all, young and old, and listen to a story that I have to tell.

Back in 2014, when we were all eagerly waiting for </SCORPION> to debut on the TV network formerly known as the Columbia Broadcasting System, a minor ruckus was raised over Junk Hacking. The moral fiber of the youth, it was said, was being corrupted by a dozen cheap Black Hat talks on popping embedded systems with old bugs from the nineties. Who among us high-brow neighbors would sully the good name of our profession by hacking an ATM that runs Windows XP, when breaking into XP is old hat?

Let's think for just a minute and consider the best examples of neighborly junk hacking. Perhaps we'll find that rather than being mere publicity stunts, junk hacking is a way to step back from the daily grind of confidential consulting work, to share nifty tricks and techniques that are often more interesting than the bug itself.

Our first example today is from everyone's favorite doctor in a track suit, Charlie Miller. If you have the misfortune of reading about his work in the lay press, you might have heard that he could blow up laptop batteries by software,¹ or that he was recklessly irresponsible by disabling the power train of a car with a reporter inside.² That is to say, from the lay press articles, you wouldn't know a damned thing about what *mechanism* he experimented with.

So please, read the fucking paper, the battery hacking paper,³ and ignore what CNN has to say on the subject. Read about how the Smart Battery Charger (SBC) is responsible for charging the battery even when the host is unresponsive, and con-

¹If you RTFP, you'll note that the Apple batteries have a separate BQ29312 Analog Frontend (AFE) to protect against such nonsense, as well as a Matsushita MU092X in case the BQ29312 isn't sufficient.

²One time, my Studebaker ran out of gas on the highway. Maybe we should start a support group?

³`unzip pocorgtfo11.pdf batteryfirmware.pdf`

⁴`unzip pocorgtfo11.pdf sluu225.pdf`

⁵`unzip pocorgtfo11.pdf bq20z80.py`

sider how much more stable this would be than giving the host responsibility for managing the state. Read about how a complete development kit is available for the platform, about how the firmware update is flashed out of order to prevent bricking the battery.

Read about how the Texas Instruments BQ20Z80 chip is a CoolRISC 816 microcontroller, which was identified by Dion Blazakis through googling opcodes when the instruction set was not documented by the manufacturer. See that its mask ROM functions are well documented in `sluu225.pdf`.⁴ Read about how code memory erases not to all ones, as most architectures would, but to `ff ff 3f` because that's a NOP instruction.

Read about how this architecture wasn't supported by IDA Pro, but that a plugin disassembler wasn't much trouble to write.⁵ Read about how instructions on the CoolRISC platform are 22 bits wide and 24-bit aligned, so code might begin at any 3-byte boundary. See how Charlie bypasses the firmware checksums in order to inject his own code.

Can you really read all thirty-eight pages without learning one new trick, without learning anything nifty? Without anything more to say than your disappointment that batteries shipped with the default password? He who has eyes to read, let him read!

Loyal readers of this journal will remember PoC||GTFO 2:4, in which Natalie Silvanovich gets remote code execution in a Tamagotchi's 6502 microcontroller through a plug-in memory chip. "Big whoop," some jerk might say, "local control of memory is getting root when you already have root!"

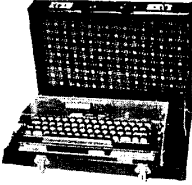
Re-read her article; it packs a hell of a lot into just two pages. The memory that she controls is just data memory, containing some fixed-size sprites and single byte describing the game that the cartridge should load. The game itself, like all other code, is already in the CPU's unwritable Mask ROM.

So given just one byte of maneuverability, Natalie tried each value, discovering that a `switch()` statement had no `default` case, so values above `0x20` would cause a reboot, while really high values, above `0xD8`, would sometimes jump the game to a valid screen.

At this point she had a good idea that she was running off the end of a jump table, but as is common in the best junk hacking, she had no copy of the code and needed an exploit to extract the code. She did, however, know from die photographs and datasheets that the chip was a GeneralPlus GPLB52X with a 6502 instruction set. So she came up with the clever trick of making a *background picture* that, when loaded into LCD RAM, would form a NOP sled into shellcode that dumped memory out of an I/O port.

By reverse engineering that memory dump, she was able to replace her hail-Mary of a NOP sled with perfectly placed, efficient shellcode containing any number of fancy new features. You can even send your Tamagotchi to 30C3, if you like.

BABY!



**Complete System
in a case!**

KEYBOARD: 62 key upper & lower case + Greek;

TAPE INTERFACE: High speed, 1200 Baud! Cassette, programs included;

VIDEO INTERFACE: E.I.A. Compatible;

MICROPROCESSOR: 6502 based system!

MEMORY: 2K or 4K byte RAM minimum system monitor + 3K ROM sockets;

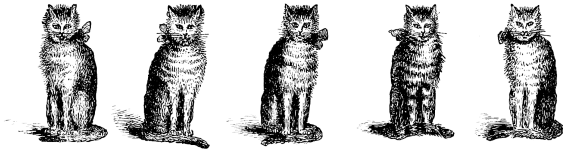
2K S.T.M. SYSTEMS INC. 4K
 \$850.00 P.O. Box 248 \$1000.00
 Mont Vernon, N.H. 03057

NOT A KIT!
BURNED IN

FULLY TESTED



The point of her paper is no more about securing the Tamagotchi than Charlie's is about securing a battery. The point of the paper is to teach the reader the *mechanism* by which she dumped the firmware, and if you can read those two pages without learning something new about exploiting a target for which you have no machine code to disassemble, you aren't really trying. He who has eyes to read, let him read!



And this is the crux of the matter, dear neighbors. We become jaded by so much garbage on TV, so much crap in the news, and so many attempts to straight-jacket the narrative of security research by the mistaken belief that it must involve security. But the very best security research *doesn't* involve security! The very best research has no CVE, demands no patch, and has no direct relation to anything from your grandmother's credit card number to your server's `shadow` file.

The very best research is that which teaches you something new about the *mechanism* by which a machine functions. It teaches you how to build something, how to break something, or how to take something apart, but most of all it teaches you how the hell that thing really works.

So to hell with the target and to hell with the reporters. Teach me how a thing works, and teach me the techniques that you needed to do something clever with it. But if you casually dismiss the clever tricks learned from hacking an Apple II, a battery, or a Tamagotchi, I'm afraid that I'll have to ask you politely, but firmly, to get the fuck out.⁶

⁶Remember, though, that redemption is for everyone, and that one day you may find a strange and radiant machine you will treasure for the cleverness of its mechanisms, no matter if others call it junk. On that day we will welcome you back in the spirit of PoC!