You Can Be a Hardware Hacker, Too!

Full Presentation Notes

Jennie Kam, @TXJennieK
Women in Cybersecurity Conference 2017
Tucson, Arizona

Introduction

These are my fairly informal notes for an introductory hardware hacking talk aimed at software engineers. I hope to appeal to the targeted audience with open source tools in higher-level languages as well as tools that will assist with on-board serial communication. In these notes, I strive to convey information, not to get a perfect score in English composition. So I will be using first person, the informal, impersonal "you," and possibly some made-up verbs.

I'm an embedded systems security researcher at Cisco. I evaluate product hardware and firmware for security vulnerabilities. Before Kali Linux and the collection of beginner friendly tools evolved, I didn't think I could conduct a port scan, exploit a cross-site scripting bug, or crack password hashes. I've had many software folks tell me that hardware is magic, which it's not, it's just electrons. I'd like to invite anyone who doesn't consider themselves a hardware person to dabble in this community with these beginner friendly tools and a few seeds of knowledge.

I've also had folks question why anyone should bother with this skillset. They make the case that hardware hacking requires physical access, and once physical access is achieved the attacker can simply unplug the target and POOF! Instant denial-of-service (or DoS.) However, there are several cases where an attacker's objective isn't a DoS. There are communities dedicated to obtaining unsigned code execution (sometimes called "jailbreaking" or "rooting") on smartphones[1][2] and gaming consoles[3][4] for a variety of motivations. In 2013, Ang Cui and Dr. Salvatore Solfo created a pluggable module that turned Cisco phones into remote eavesdropping devices[5]. In 2010, the late Barnaby Jack utilized the USB port to force an ATM dispense cash[6].

Tools for Pluggable Attacks

I'm going to define "pluggable" attacks as one where a bad actor has temporary, unsupervised access to a target and utilizes ports on the outside of the target. Think USB or thunderbolt ports on the side of a MacBook (see Figure 1).

<SOAPBOX> What I'm calling "pluggable" attacks are commonly referred to as "evil maid" attacks[7] and occasionally a researcher uses a play on words to reverse the gender ("malicious butler.[8]") But, I just didn't feel gendered domestic servant terminology deserved a place in the cybersecurity realm, especially at a women in cybersecurity conference. Won't anyone give the evil-intentioned gender-unknown interviewee some credit? </SOAPBOX>

Figure 1

USB Fuzzing

Starting with the familiar and commonly accessible: the USB port. I feel like I've noticed USB ports on many products that have no business exposing such a port (e.g. video gaming machines, parking meters.) There's a tool called the Facedancer[9] by Travis Goodspeed that allows python programmers to emulate a USB device[10]. I recommend Beyond Logic's guide[11] for USB terminology like device versus host.

Pair this with the UMAP framework[12] by Andy Davis or UMAP2 framework[13] by Cisco SAS and NCC Group for a start on analyzing USB host drivers. The UMAP framework can determine the host operating system and if the host will enumerate a variety of USB devices (as defined by class/subclass codes.) The UMAP program can also perform some automated and targeted fuzzing. It would be pertinent to note the types of devices and fuzzing supported by UMAP is not comprehensive – there is some opportunity to extend the framework.

Similar to software engineering, everything is a trade-off. While the Facedancer 21 and UMAP have decent documentation and community, if you don't have the soldering and schematic-reading skills to build one yourself for ~$35[14], it'll run you on average $100 pre-built[15,16,17].

Some other projects that experiment with USB attacks include BadUSB[18], where common USB device firmware can be overwritten to create a malicious device. While examples exist to turn a cheap flash drive (mass storage class device) into a malicious keyboard[19] (HID class device), another option is to spend some extra coin (~$45) and purchase the USB Rubber Ducky[20]. The USB Rubber Ducky Project is written in C and has good documentation[21] and community[22,23] surrounding it.

Ethernet Adapters

For the web savvy folks, I'd recommend checking out the PoisonTap[24] project, created by Samy Kamkar, which runs on the Raspberry Pi Zero[25]. It emulates a malicious Ethernet device over USB or Thunderbolt and works on most locked computers (not ones in an encrypted memory sleep mode.) PoisonTap chains a variety of exploits together to hijack all network traffic from the target computer and install a persistent backdoor by essentially man-in-the-middling the top one million common sites[26].

The trade-off here is that while Raspberry Pi Zero's can be found for as cheap as $5-10, it may cost you more in time, as the community surrounding it seems smaller.

I recently discovered the LAN Turtle[27], released just last year, by the same folks who produce the USB Rubber Ducky (Hak5[28]) that also appears to provide remote access and man-in-the middle capabilities.

Beginning with PCBs

Inside everyday electronics are printed circuit boards, or PCBs. I'm going to use the HP Veer, a credit card sized smartphone from 2011, as an example of a PCB. The Veer was one of HP's final products with PalmOS[29] before shutting it down the same year. This was also my DEFCON burner phone until HP shut down the required webOS services in 2015[30]. I took two tiny screwdrivers to the Veer – I used the Phillips to undo 4 screws and the flathead to pry off all the plastic and thin metal shielding called cans. I was not gentle, as I'm not in the business of putting things back together. You can find teardowns on the web to do this task more methodically in case reassembly is important to you.

Next look at all the plastic black rectangles/squares on the board – the integrated circuits (ICs) or informally referred to as chips. Input all the tiny text you can read off of each chip into your favorite search engine and it should be easy to identify the purpose of each one. You typically want to start with the main processor and any flash that could hold firmware. The processor is typically located close to RAM, so any chips labeled DDR1, DDR2, etc.[31] In the case of the Veer, the CPU is actually positioned underneath the mobile DRAM (LPDDR1) as a stacked ball-grid array package[32] seen in Figure 2.

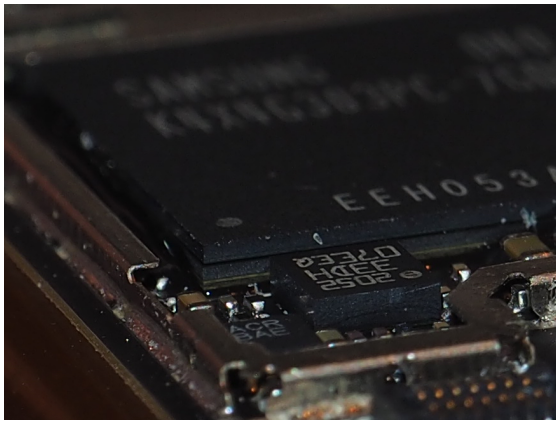

Figure 2

The Veer was actually not a great example of something to start hacking at home because the need to save space on a tightly packed PCB raised the bar for reverse engineering. The Veer was built with all ball grid array packages, which require more sophisticated equipment (or more skill) to desolder than traditional surface mount packages (aka "gull-wing.[33]") Additionally, there aren't any obvious debugging pins or test points available on the board.

Let's look at another board, which I didn't want to use in the presentation because the flash was *too* easy to read/re-write. It's on a removable microSD card (as outlined in blue in Figure 3 below!) This second board however, is a good example of potential test points

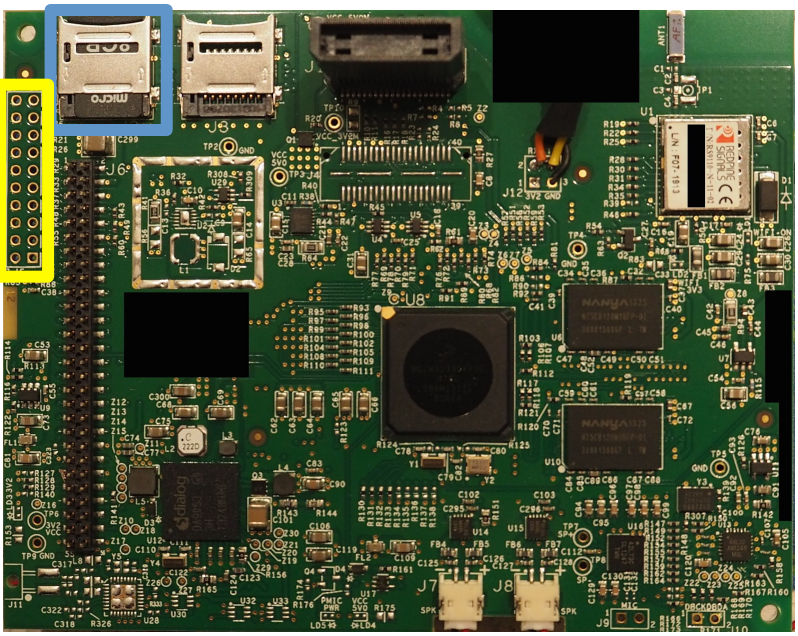

Figure 3

built-in near the firmware storage – highlighted in yellow above.

Finally a third board[34] is pictured in Figure 4 below and the debugging pins are nicely populated and labeled for us.

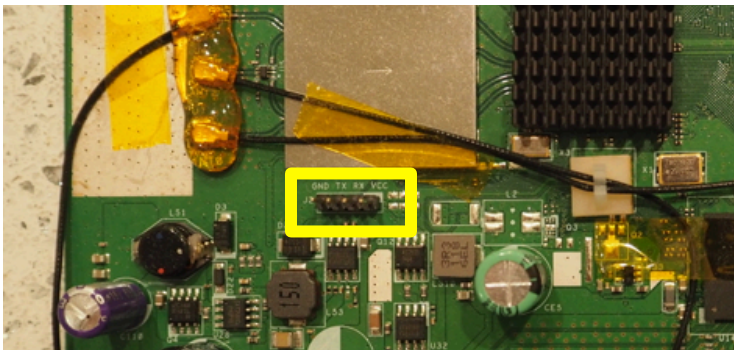

Figure 4

From left to right, you may be able to read they are GND (ground), TX (transmit), RX (receive), and VCC (power). Searching the Internet for protocols that use these pins should turn up a source on serial communication.[35]

So once you've found the debugging pins and identified the type of protocol (1-wire serial, 2-wire serial, SPI, I^2C, etc.) you're ready for a tool to help you parse what the IC is saying (i.e. monitor and decode the traffic on the bus.) The Bus Pirate[36] and Shikra[37] will come in handy here, for less than $50 each. You can pair these multi-tasking boards with the Flashrom[38] project that assists in reading nearly 500 flash chips. Once the binary has been recovered from the flash, it's ready for your favorite disassembler.

Do notice it appears Dangerous Prototypes, the original creators, have abandoned the Bus Pirate's next revision (version 4) and official firmware development. However, you can find a fairly active community surrounding this mature project (version 1 was released in 2009.) The Shikra was built to be a faster, more reliable Bus Pirate by the company Xipiter, but did not seem to be documented past the pinouts and original getting started guide[39].

The Bus Pirate and Shikra can both be used with OpenOCD[40], or Open On-Chip Debugging project, which started as Dominic Rath's thesis[41] in 2005. OpenOCD via a JTAG interface allows you full control of the CPU execution including, but not limited to, the ability to halt execution and read registers or memory as you step through a program. You might use this once you've caused a crash and need more information about the state of the processor. Or, you could use it to confirm code flow you've predicted through reverse engineering.

If test pin functionality is not clearly labeled (often you will see lonely pins marked TP1, TP2, etc.) you could break out a $170 JTAGulator[42] created by Joe Grand[43] to assist you in identifying debugging connections. The time and money trade-off appears again!

Logic Analyzers

With a little extra money, you can step up your game and get closer to the electrons in action. There are a number of USB logic analyzers (LAs) that will show you the state of multiple signals on the board at once. You typically can get insight for 4-8 signals (mostly digital) for less than $200. If you have a few processors on a board (main, Wi-Fi, cryptographic acceleration, etc.) and want to know the order in which they are provided power to boot, this would be your tool of choice. These LAs can usually decode a larger variety of on-board communication protocols than available with the Bus Pirate and Shikra. Sometimes the LAs come with firmware to write your own decoder (or dissector, as Wireshark users would call them) for proprietary protocols.

Desktop USB logic analyzers have brought the general population signal analysis abilities once available only to companies with tens of thousands of dollars to spend. Of course, there are still professional LAs that cost anywhere from $10k to $100k and can monitor hundreds of signals at ridiculously high speeds. These LAs typically also run an outdated version of Windows, but that's another topic for a different time.

Miscellaneous Advanced Ideas

Another category of attacks, side-channel attacks[44], are becoming more accessible each day through published research and tools. These types of attacks usually take advantage of the information leaked through power consumption or execution time. The Chip Whisperer Lite[45] is one of the only open source tools I know of for the average scientist to experiment with side-channel analysis. These did not make the 5-minute lightning talk cutoff because they do require some time and money investment – at least a few months and $325.

While not really a hardware attack, if you want to practice your soldering and maker skills, I recommend reproducing the Tastic RFID Thief[46] by Fran Brown demonstrated at DEFCON 21 in 2013. The Tastic RFID Thief repurposes a long range RFID reader (think those big pads you'd see in parking garages where you hold up a card within a few feet) and an Arduino to surreptitiously steal information from badges commonly used for access in corporations.

Conclusion

These are just starting points for software folks to get a taste of hardware hacking. Often a basic online tutorial of on-board protocols will give you enough knowledge to connect these tools. I encourage you to give back to this community and document your steps and missteps on the web somewhere (a public repository on Github, perhaps?) Happy (white hat) hacking, y'all.

---

[1] http://www.androidcentral.com/root
[2] https://www.theiphonewiki.com/wiki/Jailbreak
[3] http://www.ijailbreak.com/how-to/playstation-4-jailbroken-heres/
[4] http://xboxjailbreak.com/
[5] https://arstechnica.com/security/2013/01/hack-turns-the-cisco-phone-on-your-desk-into-a-remote-bugging-device/
[6] http://www.theregister.co.uk/2010/07/28/atm_hacking_demo/
[7] http://searchsecurity.techtarget.com/definition/evil-maid-attack
[8] http://www.securityweek.com/new-windows-attack-turns-evil-maid-malicious-butler
[9] http://goodfet.sourceforge.net/hardware/facedancer21/
[10] http://travisgoodspeed.blogspot.com.co/2012/07/emulating-usb-devices-with-python.html
[11] http://www.beyondlogic.org/usbnutshell/usb1.shtml#Introduction
[12] https://github.com/nccgroup/umap
[13] https://github.com/nccgroup/umap2
[14] All prices in this paper are USD.
[15] https://int3.cc/products/facedancer21
[16] https://store.hackaday.com/products/facedancer21
[17] http://hackerwarehouse.com/product/facedancer21/
[18] https://opensource.srlabs.de/projects/badusb
[19] https://null-byte.wonderhowto.com/how-to/make-your-own-bad-usb-0165419/

[20] https://hakshop.com/products/usb-rubber-ducky-deluxe
[21] http://usbrubberducky.com
[22] http://www.cryptohax.com/2015/07/dark-tips-introduction-to-usb-rubber.html
[23] https://forums.hak5.org/index.php?/forum/56-usb-rubber-ducky/
[24] https://samy.pl/poisontap/
[25] https://www.raspberrypi.org/products/pi-zero/
[26] https://aws.amazon.com/alexa-top-sites/
[27] https://lanturtle.com/
[28] https://www.hak5.org/
[29] https://www.palmsource.com/palmos/
[30] http://www.webosnation.com/hp-shutting-down-webos-cloud-services-%E2%80%94-including-backups-device-set-and-app-catalog-%E2%80%94-15-january-20
[31] https://www.transcend-info.com/Support/FAQ-296
[32] http://www.intel.com/content/dam/www/public/us/en/documents/packaging-databooks/packaging-chapter-14-databook.pdf
[33] http://www.interfacebus.com/ic-package-ic_gull_wing-drawing.html
[34] Second and third boards are purposely unnamed because the products may still be active in the market today. If you recognize them or figure them out, congrats!
[35] https://learn.sparkfun.com/tutorials/serial-communication
[36] http://dangerousprototypes.com/docs/Bus_Pirate
[37] https://int3.cc/products/the-shikra
[38] https://www.flashrom.org/Flashrom
[39] http://www.xipiter.com/musings/using-the-shikra-to-attack-embedded-systems-getting-started
[40] http://openocd.org/
[41] http://openocd.org/files/thesis.pdf
[42] http://www.grandideastudio.com/jtagulator/
[43] I credit Joe Grand's introductory hardware hacking course at BlackHat Las Vegas as the inspiration for my passion, and career path, in cybersecurity. Also a topic for another time. Maybe after an adult beverage.
[44] http://www.techdesignforums.com/practice/guides/side-channel-analysis-attacks/
[45] http://store.newae.com/chipwhisperer-lite-cw1173-two-part-version/
[46] https://www.bishopfox.com/resources/tools/rfid-hacking/attack-tools/