Toughts about Nspire « jailbreaking »

Lionel Debroux

Adrien Bertrand
(Additions + Powerpoint reformatting)

Foreword

- I'm trying to use clear, direct words, and remain in a middle ground between excessive criticism and excessive diplomacy (which are equally bad to make one's points: one turns the audience away, while the other one can mute important concerns). Offensiveness, if any, is not intended: we're adults trying to explain and discuss constructively :-)

- Throughout, I'm using "native code" instead of "assembly", as it's more precise: people can program in languages of higher level than ASM.

- Finally, writing this text (and the longer, less abstract form containing more examples, which I made first) took me ≈12h.
  Several improvements were contributed by Adrien Bertrand.

Ndless

- Ndless is the TI-Nspire incarnation of what is often termed "jailbreak" on other platforms like iPhones etc..

- "Jailbreaking" is the concept of gaining access to less restricted programming on platforms where the manufacturer doesn't provide such access, for some reason. More on that later.

- TI-68k and TI-Z80 models officially provide access to native code, which allows programmers to expand the usage envelope of calculators; games of course (it's the #1 non-math usage), but not just that.
  Programmers can (and do !) use TI-68k & TI-Z80 in creative ways.

- * Technically, Ndless is a program which exploits one or more vulnerabilities in the OS, gains control of the processor, and installs a RAM-resident program. Said RAM-resident program provides the functionality of loading native code client programs, launching them, etc.* alongside the Ndless binaries and sources, there's a SDK, based on standard open source tools (GCC + binutils) targeting the Nspire's ARM processor, and a growing library of Nspire-specific functions.

Some generalities about jailbreaking (philosophy and benefits)

- The line of thinking of many people in open development communities (not just the Nspire community) is that **manufacturers have no valid business trying to restrict programming on them**. thousands of peo find unacceptable that the manufacturer is trying to prevent them from using the devices they have bought, and which they therefore fully own, the way they see fit.

- Trying to restrict what users can do with their hardware is doomed to failure anyway: **in the past 20+ years, virtually every platform with direct access to native code has been opened up, sooner or later Almost every code / copy protection has also been eviscerated, s or later, through more or less sophisticated means.** Facts consisten show that **trying to prevent the inevitable ends up being a waste o and resources** - it only delays the inevitable.

- Throughout history, the benefits of opening devices to programming have usually been good - even for businesses !
  - Openness benefited TI Education's market share directly: in the 1990s and 2000s, many people bought TI-Z80 and TI-68k calculators for games and programs, which are a result of their programmability higher than that of comparable Casio models. This really is a fact.
  - Would TI sell as many ARM-based processors if all platforms using them were closed to programming ? Of course not. User programming is an integral part of the success of the Google Android platforms.
  - Witness the success of the Arduino micro-controller, and of NXT-enhanced LEGO: hundreds of thousands of tinkerers are using their programmability to automate lots of things in their houses, honing their skills in EE/CS/physics, and otherwise becoming the most curious and motivated developers. Just search for "Lego machine" on YouTube ;-)

Application of jailbreaking to the Nspire devices family

- The current situation for the Nspire series is that:
  - TI strongly fights attempts to unleash the power of the most powerful calculators on the marketplace, through native code;
  - just like on TI-Z80 and TI-68k calculators, TI consistently fails at efficiently protecting the Nspire – otherwise, Ndless would not be popping again every once in a while;
  - on the Nspire, TI is not happy with programmers using native code to build games instead of science programs.

- **But** in the absence of documentation, it's very much easier to make games than math programs, because the former requires much less knowledge about the platform (basically, screen, keyboard and optionally filesystem) than the latter.
  - on TI-68k/AMS, a number of math programs were written in C/ASM, precisely because it allows for faster and more powerful math programs. Using lower-level functions once enabled me to greatly reduce, in comparison to the BASIC version, the algorithmic complexity of a program implementing the Aitken delta^2 algorithm.
  - on Nspire/Phoenix, there's no official documentation about the Nspire document system, and it's hard to reverse-engineer by ourselves (which should, by the way, be considered as "reverse-engineering for interoperability purposes", exempted from DMCA prosecution), because the Nspire OS is so huge... so we can't do that.

Summary of the benefits of openness in TI calculators

- To old-timers of the TI calculators open development community, the benefits of having calculators more open to programming are clear, for both users and the manufacturer. more non-games programs, and more usage of the Nspire in fields that TI hasn't initially thought about, is seen as something that makes TI earn `more` money.
  Perhaps it's a bit simplistic of a point of view, but that's how it was 10 years ago: among thousands of persons, I bought a 89 instead of a Casio Graph 100 after seeing someone doing cool things with his 89.

- That's why we have, historically, lambasted the Nspire series so hard (in some areas, it's worse than a 20-year-old TI-81 !) and keep doing so at times. No usable graphics drawing on the Nspire has always been seen as a major blunder, all the more it hurts the Nspire's BASIC's usability (and fun factor !) in a number of classes dealing with algorithms !

Risks for TI ?

- Unrestricted programming <= "tampering with the PTT mode" ? (even if the PTT mode is not required to get a calculator certified for standardized tests: the Casio Prizm doesn't have such a special mode)

- Well, this is completely unrelated to Ndless !
  - with or without TI trying to lock down, if anyone is determined at damaging TI's Nspire business and finds the means to do so, he will succeed.
  - any arbitrary code execution vulnerability could be used for the special purpose of tampering with the PTT mode.

- Ndless <= knife: knives can be used to kill living beings... But 99.999+% of persons use knives only for the purposes of cutting food !

- Nothing catastrophic, from TI's POV, has occurred through Ndless since December 2009... though arbitrary code execution theoretically makes possible tampering with the PTT !
  The developers most knowledgeable about ways to achieve nefarious goals, are responsible human beings, fully aware of the obvious fact that such a move would actually go *against* everyone's best interests (TI would try even more actively to lock down, and we all lose).
  - even RunOS and OSLauncher, which hot-load and launch arbitrary OS (without patching them, but they could !) were not used for tampering.

- I'll argue that Ndless (unofficial) *reduces* the risk of catastrophic events occurring, and that some official openness would - besides making the Nspire more useful to users - *reduce* further the risk that "determined jerks" disrupt TI's business.

The Sony PS3 Story

- An example of what NOT to do: the Sony disaster, Sony's actions creating anger and frustration, and switching the focus of work from "making productive things" to "making destructive things" (you can cross-check my description on the Internet, thousands of pages mention the events):
  - first, a while ago, Sony abruptly closed access to Linux on the PlayStation 3 (PS3), after providing it for years. Before that, for 4 years, tinkerers were happy, and no reverse-engineering that was a real threat to Sony took place;
  - later, they sent powerful legal attacks to some tinkerers, who had been performing reverse-engineering with a new goal of (at least)gaining back what Sony gave and took away, and possibly getting complete control of the PS3 (which is, again, a basic right of users).
  - very quickly after Sony's legal attacks, the Anonymous, a powerful online group of ultra-skilled crackers, thoroughly penetrated Sony's infrastructure and disrupted the PlayStation Network, knocking it offline for days (multiple times), causing Sony hundreds of millions of dollars (!!) of immediate losses (and unspecified future losses due to the PR disaster and inconveniencing of gamers) !

The Sony PS3 Story

Comments:

- The PS3 castle fell after merely one year of people trying to look at it with a different focus, after growing peacefully for 4years before that - people weren't even trying to damage it, since they had something which they deemed "good enough". This used to be, again, responsible behavior from the technically knowledgeable persons - until Sony started behaving destructively.

- Of course, I'm not saying that anyone from the TI community will ask the Anonymous to target TI, for a ghastly revenge, if TI doesn't provide more openness for the Nspire ;-) We're responsible persons, and for one thing, nobody (besides Adrien, Jérémy and Xavier) knows that I'm writing this text.

Moving Forward ?

- The current status quo can go on for years, if nobody gets sufficiently both skilled and motivated to damage TI's business:
  - TI fighting (e.g. through more complicated steps used on the CX & CM to obfuscate the OS), people defeating them (said complication was just a minor annoyance) and opening the calculators to a short flurry of development for games, until the next OS upgrade which closes the holes;
  - no native code math programs, because it's very hard to make them.

- BUT we could do better than this, through less hostility from the manufacturer (i.e. TI behaving more like in the past with older models, in fact) ;-)

- If TI wants more math & science programs (and the faster, more powerful math manipulation which native code can provide to skilled programmers), at a minimum, we need to be able to:
  - integrate both ways with Calculator screens (similar to what Nspire BASIC libraries can do: take arguments, return values);
  - know how to read/write per-document variables;
  - know how to make a native code program well-formed for the purposes of containing per-document variables.

General Note

- The community's strength comes precisely from its unity. Those who produce little useful content by themselves, but try to divide and conquer the community to further their own "special" agendas - yes, I'm referring to an extremely destructive person we mentioned in the Paris meeting, whose behavior keeps worsening - are bad for the community…

- Well, this wraps up this lengthy writing. I tried to do my best to expose the different points of view, especially the ones "from the other side", which you probably don't hear about every day :-)

Thanks for reading