3. also allow the user to reliably write content to the stick (e.g. an image for a trustworthy BIOS the user decides to use).

The Embedded Controller's flash memory

The Embedded Controller (which should not be confused with Intel ME) is a little auxiliary, discrete, microcontroller connected through an LPC bus to the chipset. It is responsible for 1) keyboard operation, 2) thermal management, 3) battery charging control and 4) various other OEM-specific things, such as LEDs, custom switches, etc. More discussion about how this uC might be compromising platform's safety has been provided in [10].

In this paper, however, we're more concerned with the fact that the uC that is used to implement EC would typically also contain an internal flash memory (e.g. see [5]), yielding it a state-carrying element on the platform, something we would like to avoid.[7]

We thus would like to use a uC without a built-in flash-able memory, one that expects the firmware for execution to be provided by an external chip. One example of such a chip is the one used by the OLPC and Purism laptops [7].

The hard disk

The internal hard disk is an obvious device which is capable of storing the state. In fact this is the very reason disks are made.

What might be less obvious though, is that disks typically contain their own uC with their own internal flash-able memory. This naturally breaks the stateless requirement for the platform even further...

Also, due to potentially backdoored firmware, or just due to how modern solid-state disks work (wear-leveling mechanisms), some information stored directly on the disk by malware (such as the stolen user disk encryption key) might not be easy for the user to wipe using traditional disk shredding methods.

[7] Admittedly the EC, no matter how evil firmware it executes, would not be able to interfere with the platform boot sequence, and thus would not be able to compromise the system or any other software execution directly. However, as already discussed in [10], the EC might pull out a few other, more subtle attacks, such as e.g. injecting keystrokes that could trigger some actions that might also be fatal. Or, as one of the reviewers noted, might pretend the system is off when it really is not, which might be problematic e.g. when switching between supposedly separate boot environments, or trying to prevent potential Cold Boot attacks.