AmberWolf

Thin Client? Thin Crypto: An overview.

Darren McDonald | 08 August 2026

Summary

Back in February 2025 I found a simple bug, a thin client operating system that saved its crash dumps to the disk unencrypted. Shocked by what I saw, I started looking deeper and found a very suspicious looking Key Derivation Function (KDF) protecting the disk encryption. I pulled on that thread for 18 months, and ended up finding a pile of bugs big enough that it got me a speaking slot at DEF CON 34 main track. This blog post is an overview of what happened, and where it leaves thin client full disk encryption in general.

The short version of what I discovered is that every one of the three vendors that dominate the thin-client market, HP, Dell and IGEL, shipped full disk encryption you can defeat. As of early 2026 that was true across all three at once. The complexity of these attacks was fairly low, not at all what the phrase “TPM attack” usually conjures. No bus-sniffing, no glitching, no cryptanalysis. It was mostly reading bash scripts and using a USB stick.

The five vulnerabilities

- HP ThinPro 8 & 9 seals the LUKS key to PCRs that measure the platform firmware and the GRUB binary. The kernel and initramfs are unencrypted and unmeasured, so you patch the initramfs to dump the key after the TPM releases it. This issue is not patched at the time of writing.
- Dell ThinOS 9.x (FreeBSD, Geli) seals a random key in the TPM, but the password that unlocks it is derived from the device's own serial numbers, CPUID, and MAC address, all of which are printed on the label or can be determined from looking at hardware components visually. Easier still, with no Secure Boot and no PCR binding, you can just patch the boot partition and have the device give you the key. ThinOS 9.x is now unsupported and Dell will not be patching it, so this issue is permanent.
- Dell ThinOS 10.x (Ubuntu, LUKS) is a complete rewrite that came out weaker: the LUKS key sits in a TPM slot with no password and no policy, readable by any OS you boot. Dell patched it to bind the key to PCR 7, but I bypassed it the morning after the advisory was released because PCR 7 measures who signed the booted code, not what the code is, and Dell uses a bootloader with Canonical's stock Ubuntu key. A plain Ubuntu USB collides with the same PCR 7 value, and drops you into a root shell where you can just ask the TPM for the key.
- IGEL OS 12, first bypass: in TPM+PCR mode with Secure Boot, the signed bootloader reads an unencrypted, unsigned configuration area and injects its contents into the kernel command line. Write a Linux loader parameter there, reboot, get a root shell with everything mounted. The TPM unseals normally because the attack doesn't modify the trusted code; the trusted code is the problem.
- IGEL OS 12, second bypass: drop an unsigned and empty file called `igel.conf` onto a partition and GRUB fails open trying to verify it, dropping into an interactive grub prompt. From there you boot the device's kernel with some additional arguments that drop you into a root shell with the disk unlocked. Same binaries, same measured PCRs, so the TPM unseals.

How did this happen?

These are five vulnerabilities which show how TPM based full disk encryption can fail. The TPM is not bypassed; mostly what happened here is that the TPM did not measure the things that allowed control of the boot process. In the case of IGEL that was 256kb of boot registry values, HP only measured half the boot process, and in the case of ThinOS 9 and 10 nothing about the boot process was measured at all.

These attacks were not particularly clever, and they were caused by decisions that come across as obviously bad ideas:

- Trying to make an algorithm secret instead of the key material doesn't work (ThinOS 9)
- Using a security device without using any of its security features (ThinOS 10)
- Executing unverified code is a bad idea (HP ThinPro 8 & 9, Dell ThinOS 9 & 10)
- Ingesting untrusted data into a boot process is a bad idea (IGEL 11 & 12)
- Systems should fail closed if they encounter an error during a security-sensitive operation (IGEL 11 & 12)
- Small engineering teams rolling their own full disk encryption (IGEL, HP, and Dell)

These are choices, not bugs

It is tempting to read these five bypasses as five mistakes, implementation issues of the sort a patch cleans up. But to me these look less like mistakes and more like business decisions. I've spoken to various current and ex-employees of these organisations over the years and they all came across as talented and driven people. I would like to believe someone inside these organisations would have been smart enough to understand the risks and said "this is a bad idea, this isn't how we should be doing this".

In each case a secure option existed and the vendor picked another. Dell's FDE strategy seems to be absolutely allergic to the idea of going through the protracted process of getting their own signing key. That only leaves options which are either brittle (like the PCR 1+7+9 they ended up with after patching) and risk bricking endpoints, or PCR policies that do not work.

HP have been through the Red Hat shim review board, got their signing keys, then didn't finish the job and only measured half the boot chain, leaving large parts of the boot process unprotected. As of writing HP have not shipped a fix.

IGEL lets a signed bootloader trust an unsigned configuration area for pulling grub linux loader parameters, because signing and measuring that area securely is difficult if a system administrator also has to edit it. The custom boot command feature is just incompatible with the idea of secure boot and full disk encryption. The 2nd IGEL vulnerability does however seem like an oversight.

None of these is a slipped bracket or an off-by-one, these appear to be mostly design decisions which deprioritised security. What's worse, the right answer was already in the box and the same call was made independently by three vendors over more than a decade.

This is a solved problem

All of these organisations have engineered their own full disk encryption. But the standard answer, the one that's had more than two pairs of eyeballs on it, already exists and has done for years. A unified kernel image bundling the kernel, initramfs, and command line into one signed binary, with the LUKS key enrolled via `systemd-cryptenroll` and bound to PCR 7 *and* 11, optionally behind a signed PCR policy and a PIN. PCR 11 measures the actual boot payload, so any modification to the kernel, initramfs, or command line changes the seal and the TPM refuses to release the key. It closes the gap that four of these five bypasses depend on.

Dell's ThinOS 10.x is Ubuntu-based and ships `systemd-cryptenroll` on the image, unused. HP ThinPro 8.1 ships it too, at `/usr/bin/systemd-cryptenroll`, unused. Each vendor instead wrote custom scripts and custom key handling, and each custom implementation got the binding wrong in its own way.

The "physical access" vector makes this a "medium"

One objection I'm anticipating to all of this is that it needs physical access, and it's already game over if someone can touch it. This is shown clearly in the CVSS scores for all of these bugs, which land in "medium" because their Attack Vector is Physical (`AV:P`). No one is going to be exploiting this en masse over the internet, and how is an attacker going to get physical access to these devices anyway?

But during this project I was exposed to credential material and other secrets for various organisations without having to physically enter their premises. How? I bought their thin clients from refurbishers off eBay. The platform design of thin clients makes them challenging to dispose of securely. Many of these devices do not have M.2 slots, and instead use an eMMC storage chip soldered directly onto the motherboard. Technically these could be removed, and while people like myself have the skills and equipment to desolder these chips (and the resulting scars from hot plates), it is absolutely not practical for a fleet of thousands of thin clients. So many of these devices are resold without their storage being separated from their TPM chips and securely wiped, something that usually kills any attempt to recover data from a device in the secondary market.

Full disk encryption exists to stop exactly these kinds of problems. It's designed to operate in scenarios where the adversary has the hardware, and the full disk encryption is the control that is supposed to make the data worthless to them. "Physical access means you have lost" is a way of saying disk encryption has no purpose. For one single device the risk is perhaps *medium*, but for a CISO responsible for 20,000 devices

spread across multiple countries the risk that one of these devices falls into the wrong hands becomes a certainty.

Where does this leave thin clients?

As of publication, 8th August 2026, most of these bugs are patched. HP state their fix is still stuck in QA and blocked from release. Dell will not be patching ThinOS 9, and large numbers of ThinOS users didn't seem to get the memo that it recently became unsupported. Unfortunately, the kind of organisations that use thin clients tend to be big financial services organisations, healthcare, CNI, and government. Not sectors known for being able to respond quickly to changing circumstances.

I found a mechanism to bypass the TPM security and gain access to the encrypted file system for every thin client operating system I looked at. There was no thin client operating system I looked at and failed to break. Likewise in each case I only found as many vulnerabilities as I felt I needed to prove my point. Thin clients have proved a particularly soft target when it comes to full disk encryption security, and I suggest it is highly likely that, with more time and effort, someone with a modest offensive security skillset could find additional vulnerabilities.

The security of these products has been improved through recent remediation efforts in response to my work, but this is still a relatively unexplored area of security research. Though I hope my own work will encourage others to look. Which leads us to our next problem.

Lessons for the security industry

The uncomfortable thought is not that one vendor got this wrong. All three did, and have for many years. IGEL had the best architecture of the set, TPM and PCR with Secure Boot genuinely on, and it fell to the same class of problem as the other two.

I went back to Dell ThinOS 8 for context, originally released in 2014 and EoL as of 2024, and static analysis showed similar problems, down to a crude and totally ineffective attempt to use PCR 16, a debug register. Other researchers have looked at thin clients extensively and found plenty of bugs, but too few took the time to look at one of these devices' key security features, myself included; 2025 was not the first time I'd looked at a thin client. So the flaws have sat there for all this time, pentesters and researchers crawling over the platforms, all somehow missing that the disk encryption was broken.

So we have a class of fairly obvious bugs going unnoticed by the security industry, across multiple vendors (and probably more than the ones discussed here), for over a decade. And not some super subtle implementation flaw either. It is worth pointing out that most of the organisations that run thin clients are exactly the kinds of organisations that pay for professional pentesting services.

I put that down to many offensive security professionals being put off looking at bootloaders and TPMs; 8 years ago, when my work was mostly lying to banking staff to get them to open macro-enabled spreadsheets, I would have definitely included myself in that camp. From the outside TPMs look like complex, unknowable systems that require deep lore to understand. It feels like it would take a career's worth of experience to understand. While there is definitely some truth to that in how the firmware and hardware works, other parts of these processes extend into more familiar parts of the operating system and bootloader, and can be attacked using far simpler methods.

Final Takeaway

For the vendors: consider if you should be rolling your own TPM key handling and sealing out of the public with a small engineering team. To get this right you either need to throw large amounts of resources at it, or do it in public in collaboration with others. Bind to PCRs that measure all parts of the boot payload, do not derive secrets from values printed on the chassis, and enforce Secure Boot on. The pre-existing tooling already does all of this, it's open source so you could literally copy it.

For defenders running these fleets: patch, but do not assume the encryption protects a device that has left your control. Set BIOS passwords and enable Secure Boot to somewhat raise the bar, and consider physically destroying storage on disposal if you have high security requirements rather than returning or reselling units. If your platform supports it, consider adding a PIN as this would dramatically cut your risk.

For everyone in offensive security: the next time a product says “TPM-backed full disk encryption”, the first question you should ask yourself has nothing to do with the cipher, the firmware, or the hardware. Instead ask “what exactly was the TPM told to check, and is it enough?”

Further reading from AmberWolf

A full write up is coming for each issue, starting with ThinOS 10.

- Dell ThinOS 10.x: the key under the door mat, and the patch that trusted the wrong certificate
- HP ThinPro 8.1: TPM-sealed disk encryption defeated by an unmeasured initramfs
- Dell ThinOS 9.x: TPM-sealed disk encryption unlocked with the serial number
- IGEL OS 12: defeating TPM+PCR encryption via the boot registry
- IGEL OS 12: a root shell by crashing GRUB with an unsigned file

Vendor Advisories

IGEL

- ISN-2026-20
- ISN-2026-19

HP

- No public advisory published at the time of writing.

Dell

- DSA-2026-214
- DSA-2026-300

Disclosure

All of this was reported to the vendors under coordinated disclosure between February and April of 2026, and the 90-day windows have elapsed.

Where CVEs have not been issued by vendors, I have requested to reserve CVEs but have not yet received them.

Vulnerability **Dell** **HP** **IGEL** **TPM** **Encryption** **Defcon**



Written By

Darren McDonald

Red Team Operator @ AmberWolf

You May Also Like

Darren McDonald 08 Aug, 2026

#Vulnerability #Dell #TPM #Encryption #Defcon

Dell ThinOS 10.x: the worst use of a TPM possible

A SPI Flash would have at least been cheaper.

Read Article ›

Darren McDonald 08 Aug, 2026

#Vulnerability #HP #TPM #Encryption #Defcon

HP ThinPro: TPM-Sealed Disk Encryption that only measured half the boot chain

Measuring only half the boot chain doesn’t help.

Read Article ›



Copyright AmberWolf 2024-2026