Exhibit A: Partners in Bootcrime

UEFI, HP, and Me:
A Bootloader Love-Hate Story

How one stubborn laptop taught me more about firmware psychology than I ever wanted to know

This is a story about this lovely old HP business laptop and its “interpretation” of the UEFI specification -- “Quirky” would be generous.

Exhibit B (i):

Demystifying Bootloaders

Term	Meaning	Reality
BIOS	*Basic Input/Output System:* A ghost of the original IBM PC/XT	Bodges layered upon bodges; boots your hot new PC as if it was fresh out of 1983
UEFI	*Unified Extensible Firmware Interface:* The modern replacement	A fresh take; smarter, flexible, but still occasionally cursed
CSM	*Compatibility Support Module:* A component of UEFI that allows legacy booting (a.k.a BIOS mode)	Usually throws away UEFI's ability to recognize a GPT disk set up for legacy booting, falling back to BIOS heuristics for determining bootability
MBR	*Master Boot Record:* A 512-byte relic that supports, without hacks, a maximum of 4 partitions	Designed when a "large drive" was 10MB; yet somehow still alive
GPT	*GUID Partition Table:* Modern, robust, and sanity-preserving	The only sane partitioning scheme for 2025; theoretically supports BIOS booting

There are two ways you can boot a modern PC.

You can pretend it's still 1983 and boot your super-duper **Ryzen Threadripper 7995WX** exactly like it's a **4.77 MHz Intel 8088**. **BIOS** (or **CSM**) will dutifully set up a handful of abstraction layers (in 16-bit real mode, mind you) to make your modern computer look like an IBM 5150, load the first 512-byte sector of your **8 TiB NVMe SSD** into a sacred address somewhere in the first 640 kiB of your **128 GiB of DDR5**; then plunge head-first into it – still in real mode, still limited to a 64 kiB code segment – and hope for the best.

Or, you can do it the sane way; with **UEFI**, which actually understands partition tables and filesystems, natively supports USB and PCIe, can load drivers, draw graphics, run applications, and directly boot your operating system kernel with the CPU already in **64-bit protected mode** where it belongs.

Naturally, UEFI works equally well with both MBR and the preferred GPT partitioning scheme.

BIOS works best with MBR-partitioned disks, but can be coerced into booting GPT-partitioned disks. However, actually doing so is akin to firmware necromancy – and best avoided unless you *really* enjoy debugging ancient rituals; as I was to discover when I started poking at Haiku for my previous talk. But, more on that later.

Exhibit B (ii):

Meet the Stubborn Hardware

- **HP Elitebook 8560p (circa 2011)**
- **Early Insyde-based EFI firmware**
- **EFI support primarily intended for internal consumption (e.g. diagnostics and updates)**
- **Dual-drive setup: Crucial MX200 mSATA SSD (Expansion bay) + original WD Black HDD, both 500 GB**
- **Mission: A modern, reliable boot into any OS**

This laptop comes from that awkward moment in history where OEMs were only just coming to terms with EFI on the PC platform – Of course, HP had been doing it for *years* in the server space; they really had no excuse for the horrors we are about to witness.

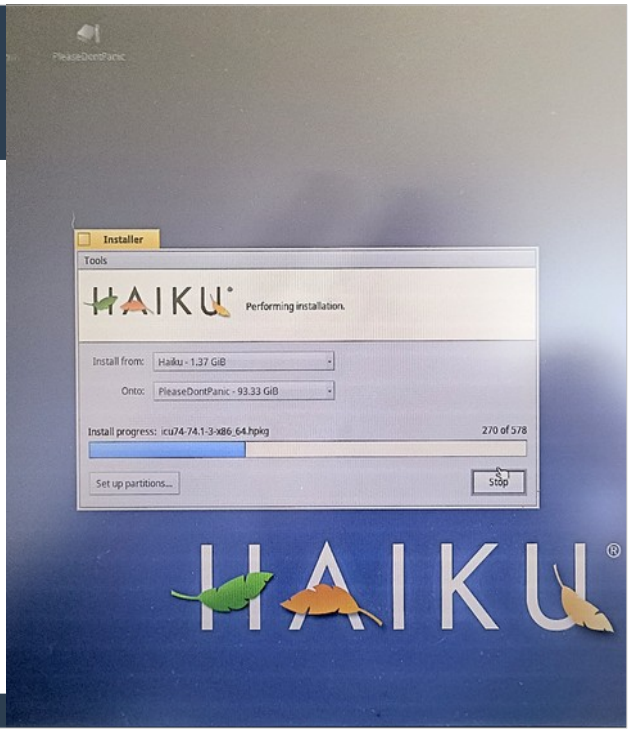
Regardless, UEFI of this time was mostly used for *vendor tooling*; it wasn't actually expected that end-users would *use* it.

Aside: The astute among you may ask; why mount the SSD in the expansion bay (where it is throttled to SATA II), when it could be running SATA III in the HDD bay? That's a reasonable question, and it originally came down to ease of mounting; but I have since rectified this with a custom mounting bracket.

Exhibit C: Legacy Booting GPT

The Catalyst

- **GPT disks booted in legacy mode**
- **Fragile BIOSBOOT shim prone to breakage**
- **GRUB forgot how to boot Haiku (after convincing it to reinstall)**
- **How hard can it possibly be to boot UEFI?**



Legacy booting of GPT disks is *technically* supported; and there is no sane reason to use MBR partitioning in 2025; so that is exactly what I did.

It would've worked, if it weren't for those meddling partition managers. You see, the majority of late pre-EFI BIOSes (and, apparently, some early UEFIs that really should know better) will refuse to boot a disk unless its MBR partition table – which lives on as part of the GPT header – has the "boot" flag set.

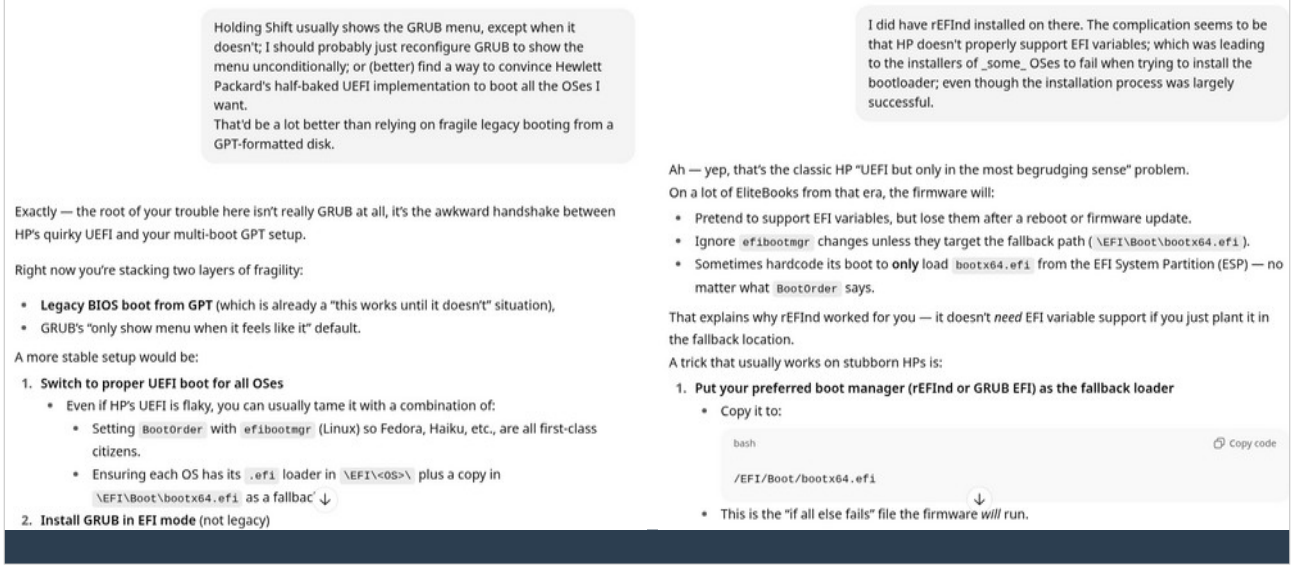
Now, in theory, this is fine. GPT specifies a "BIOS BOOT" partition type specifically for this purpose, and the boot flag in the MBR can redirect boot to that just fine. And **grub2-install** will even set things up this way if it detects the appropriate partition layout.

Unfortunately, on paper, virtually every GPT partitioning tool out there completely ignores the BIOS BOOT partition and nukes the "boot" flag from the protective MBR; rendering the disk unbootable. There is a workaround for this, of course – "Hybrid MBR"; which is even more fragile than it sounds.

So, there's really only one option here. I need to convince the firmware to boot its modern disks using modern UEFI.

Exhibit D: The Accomplice

“Sure, I can help you with that! Absolutely!”



So, I asked GPT-5 for advice. It seemed like the right thing to do; considering it had just finished a post-mortem on my Haiku install; which boiled down to a corrupt BFS partition and a BFS formatting tool saying “I made a new filesystem; she’s all good, bro” while ignoring the leftover landmines the installer and its kernel was repeatedly tripping over.

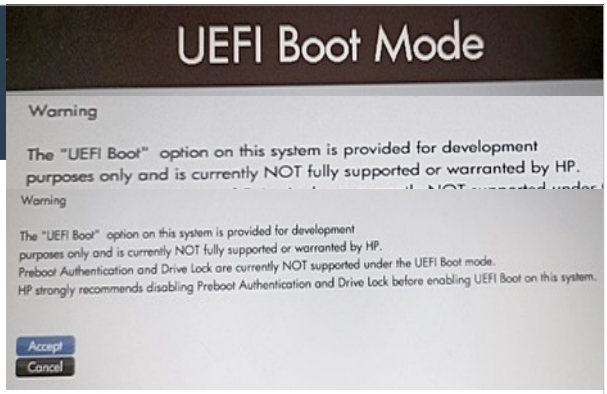
This began a journey of mutual discovery – mostly discovering how badly HP can engineer UEFI; and had the side-effect of actually documenting the journey so I can talk about it today.

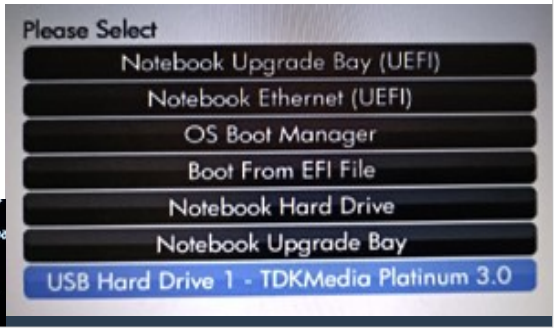
It also birthed a several-month-long side project just so I can edit **refind.conf**; but that’s a topic for another day.

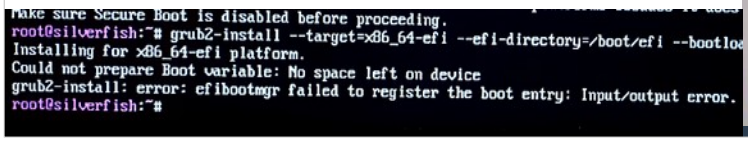
Exhibit E: Breaking Bootvars

The Observation

- **Scary warning when enabling UEFI**
- **Installing OS in UEFI mostly works, but can fail when setting bootvars**
- **HP ignores bootvars that are set**
- **CSM cannot be disabled**
- **“Boot From EFI File” is the only reliable way to EFI boot external media**





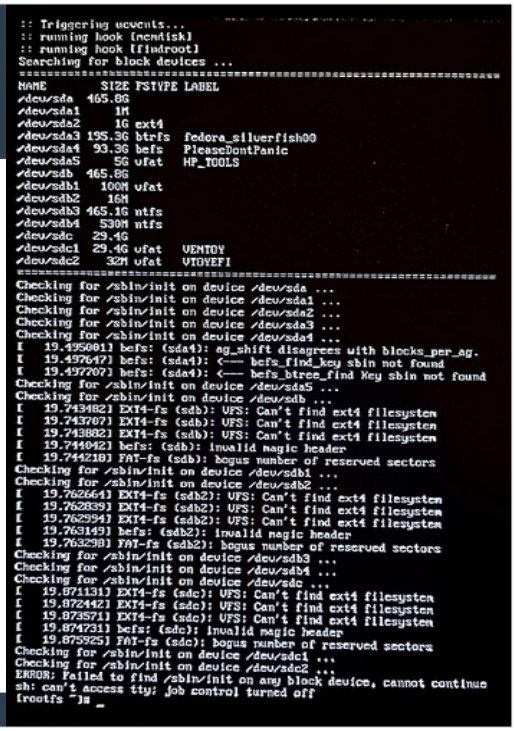


It’s EFI, Jim; but not as we know it. Intimidating warnings, mysterious OS installation failures, and a firmware that goes all in on pretending it’s nothing more than a highly evolved BIOS wearing an EFI costume. It gestures enthusiastically toward UEFI, while fundamentally refusing to cooperate.

Exhibit F:

The Plan

- **Boot Fedora in EFI mode using SystemRescueCD**
- **Install rEFInd and an EFI bootloader**
- **¯_(ツ)_/¯**
- **Profit**



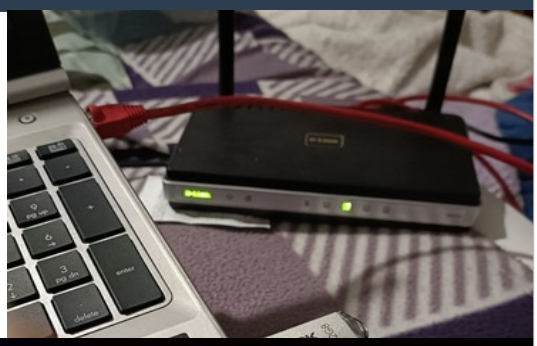
The plan was simple. Use a rescue disk to boot my Fedora installation in EFI mode, install its favoured EFI bootloader (GRUB? Systemd-Boot? EFI-stub? I wasn't going to be picky), along with rEFInd as a nice boot selector, *shrug*, and profit.

So, I found a rescue OS, and dived headfirst into attempting to boot Fedora in EFI mode so I could install an EFI bootloader for it. This went about as well as expected.

Exhibit G: Kernel Out-of-Body Experiences

The Reality

- **After much ado, Wayland boots**
- **But no mouse, no WiFi**
- **Switch to VT, realise half of PCI devices missing, fall back to wired Ethernet**
- **Download & install GRUB EFI and rEFInd**



```
root@silverfish:~# grub2-install --target=x86_64-efi --efi-directory=/boot/efi --bootloader-id=Fedora
Installing for x86_64-efi platform.
grub2-install: error: This utility should not be used for EFI platforms because it does not support UEFI Secure Boot. If y
Make sure Secure Boot is disabled before proceeding.
root@silverfish:~# grub2-install --target=x86_64-efi --efi-directory=/boot/efi --bootloader-id=Fedora --force
Installing for x86_64-efi platform.
Could not prepare Boot variable: No space left on device
grub2-install: error: efibootmgr failed to register the boot entry: Input/output error.
root@silverfish:~#
```

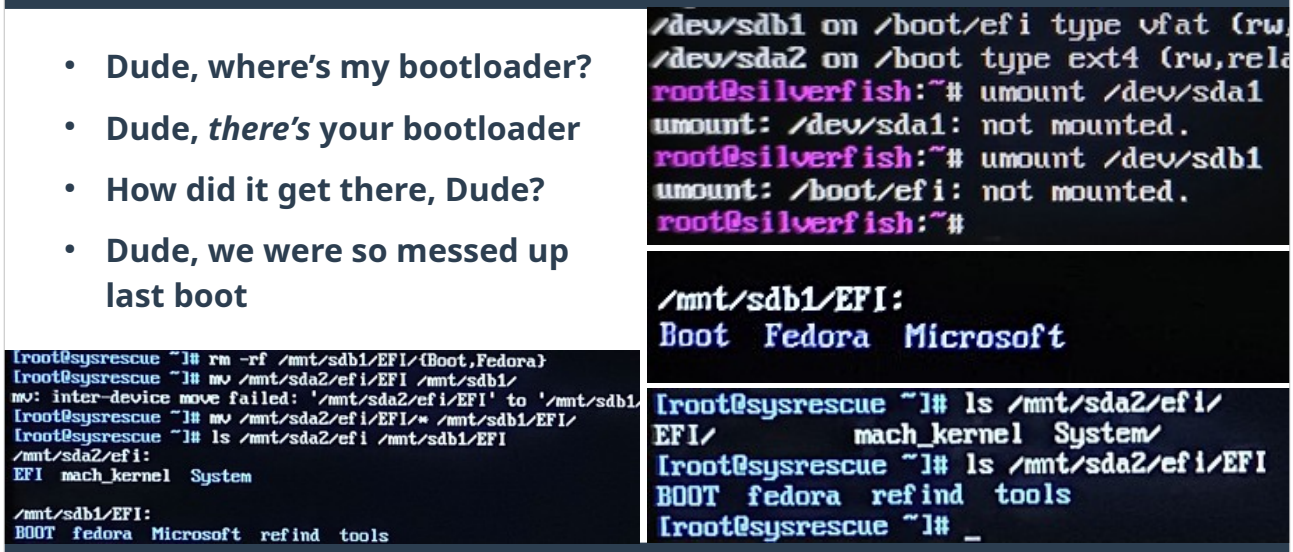
Regardless, I eventually stumbled upon the magic incantation to mount Fedora's root inside SystemRescue's initramfs and boot into it. Eventually, I was greeted with SDDM on Wayland, and I thought it would be smooth sailing from here. I was wrong.

Fortunately, SystemRescue's kernel has an Intel 82579LM driver that works. And fortunately, I have this little D-Link router I routinely use as a reverse WiFi access point. So, I proceed to install GRUB and rEFInd to the mounted Windows ESP on the SSD. Or, so I thought…

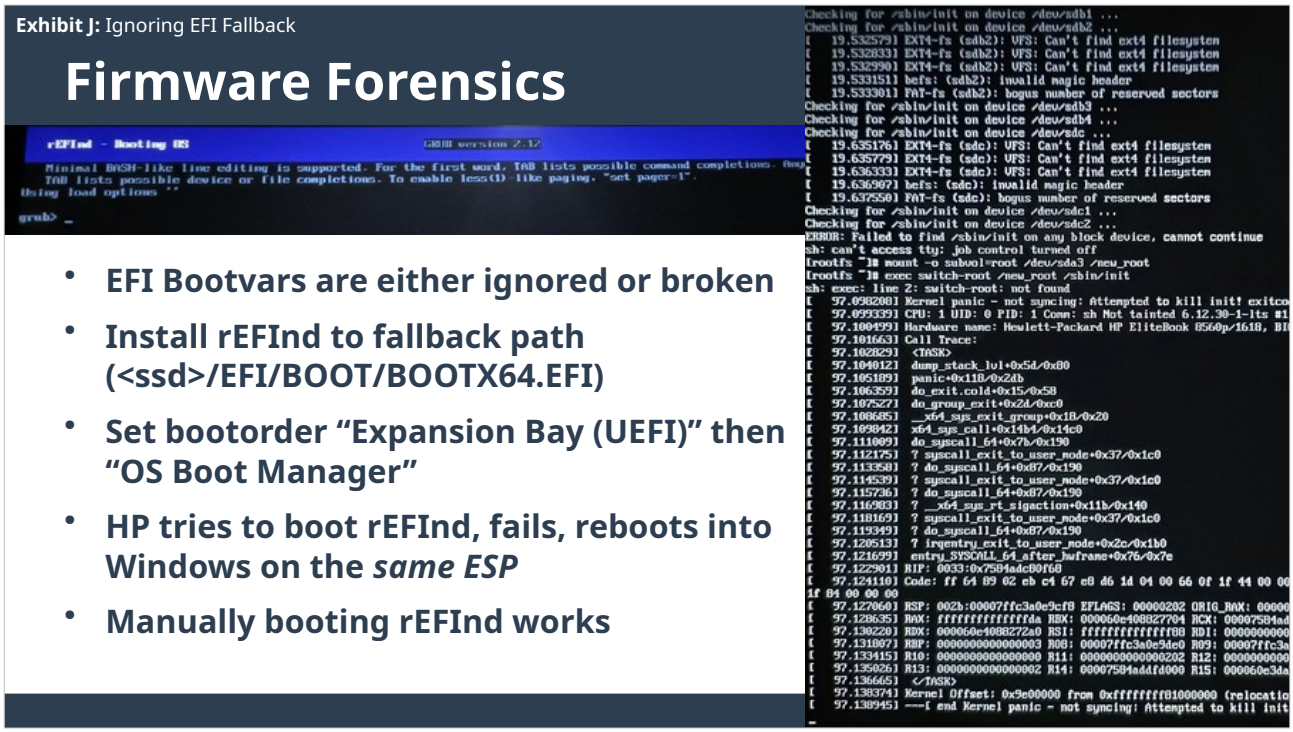
Exhibit H:

Bootloader Hide-and-Seek

- **Dude, where's my bootloader?**
- **Dude, *there's* your bootloader**
- **How did it get there, Dude?**
- **Dude, we were so messed up last boot**



As it turned out, "mount" had completely lost its sanity and failed to actually mount the ESP; so all those boot files ended up in a literal "EFI" directory inside Fedora's ext4 boot partition. So I put them where they belong and rebooted.



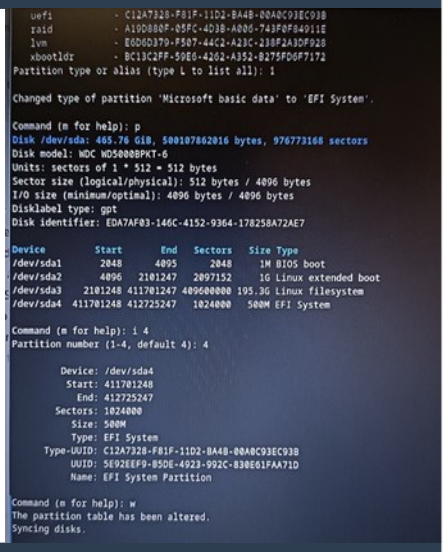
After *definitely* not forgetting to regenerate grub.cfg, hijacking SystemRescue’s initramfs to boot back into Fedora, and installing GRUB *properly*; it was time to dive deeper down the rabbit hole. rEFInd and Fedora now boot; but only if I directly point the firmware to /EFI/BOOT/BOOTX64.EFI on the SSD’s ESP.

But, I ain’t going to use the equivalent of a “File→Open” dialogue to boot an operating system – that’s insane!

Exhibit K: Ignoring ESP Markers

HP_TOOLS Experiment (Pt. 1)

- **Try to repurpose “HP_TOOLS” at end of HDD as ESP**
  - Install rEFInd to “/EFI/BOOT/BOOTX64.EFI”
  - Firmware ignores it (not in bootorder selection) and continues booting Windows
- **Use partitioner to “bless” HP_TOOLS as a real ESP**
  - Firmware still stubbornly refuses to acknowledge it
- **Shrink from 5GB to 500MB?**
  - No change in behaviour
- **Rename from “HP_TOOLS” to “ESP_HPTOOLS”?**
  - Completely indifferent



The Windows utility I originally used to create the “HP_TOOLS” partition and install the firmware update and diagnostics decided it should be 5GB. Why? Who knows. So, I figured I might as well try using it as an ESP. This was met with *some resistance*.

So, I used fdisk and parted to “bless” it as a System ESP. Still no.

Let’s shrink it. Maybe it’s confused by a 5GB ESP? No.

Maybe it’s refusing because of the filesystem label? No.

Exhibit L: Vendor Favouritism

HP_TOOLS Experiment (Pt. 2)

- **Rename “HP_TOOLS” to “ESP_HPTOOLS”**
  - Breaks connection to HP diagnostics
  - Still refuses to enumerate HDD as an EFI boot candidate
- **Copy ESP contents from SSD into “ESP_HPTOOLS”, pull SSD**
  - Firmware *still* refuses to acknowledge it as boot option
  - No problem finding Windows Boot Manager, though
- **Rename “ESP_HPTOOLS” back to “HP_TOOLS”**
  - Same result, but tools work
- **Remove Windows Boot Manager from “HP_TOOLS”, masquerade rEFInd as bootmgfw.efi**
  - *Finally,* rEFInd boots by default
- **Chaos theory: Reinstall SSD to expansion bay, with authentic ESP and Windows bootloader**

Renaming the partition broke its connection to HP diagnostics, but otherwise had no effect on the firmware’s boot behaviour.

So I copied everything from the SSD’s ESP onto the HP_TOOLS partition, removed the expansion bay, and tried again.

This was when GPT and I came to the painful realisation that HP’s firmware was really only ever looking for *one* thing; and it was neither EFI boot variables nor standard EFI fallback paths.

So, I begrudgingly installed rEFInd as “/EFI/Microsoft/Boot/bootmgfw.efi” and cursed VFAT for not letting me symlink to it. *Victory by deception*.

Exhibit M:

Current Situation

- **“HP_TOOLS” resides as first partition of first disk, as an ordinary “msftdata” partition**
  - Contains rEFInd at “/EFI/Microsoft/Boot/bootmgfw.efi”
  - Contains HP Tools at “/Hewlett-Packard”
- **“EFI_SYS” is next partition, tagged as real ESP**
  - Contains all OS bootloaders, including Windows
- **Firmware doesn’t care about special flags or partition names; just boots the first “bootmgfw.efi” from the first FAT32 partition it sees**

The most cursed thing about all of this? HP literally doesn’t care about the partition type; well, at least, not on GPT. If it’s a FAT32 partition, and it contains /EFI/Microsoft/Boot/bootmgfw.efi; then it’s booting it, no questions asked.

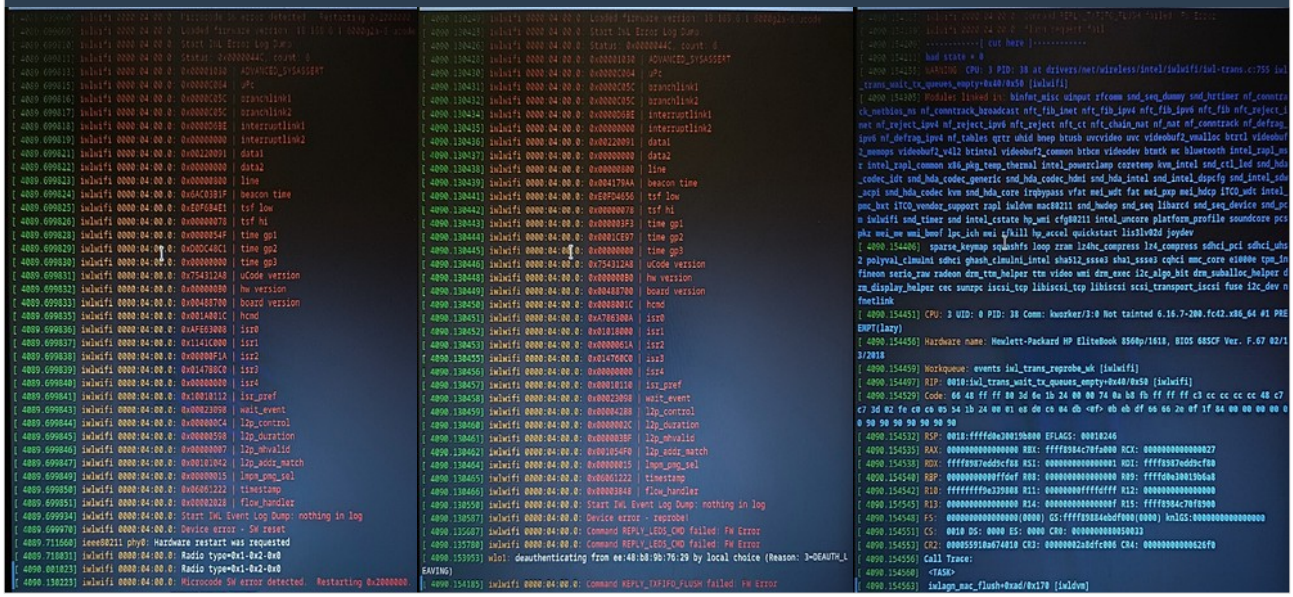
So, now I have:

* A perfectly ordinary, innocuous-looking data partition that’s secretly an ESP with rEFInd as a trojan horse.
* Next door to that, a real ESP, containing real bootloaders for real OSes
* A UEFI-ish firmware that is blissfully unaware of all the chaos it has created.

I *wish* that was the end of the story; but the worst is yet to come.

Exhibit OMG:

WiFi Woes



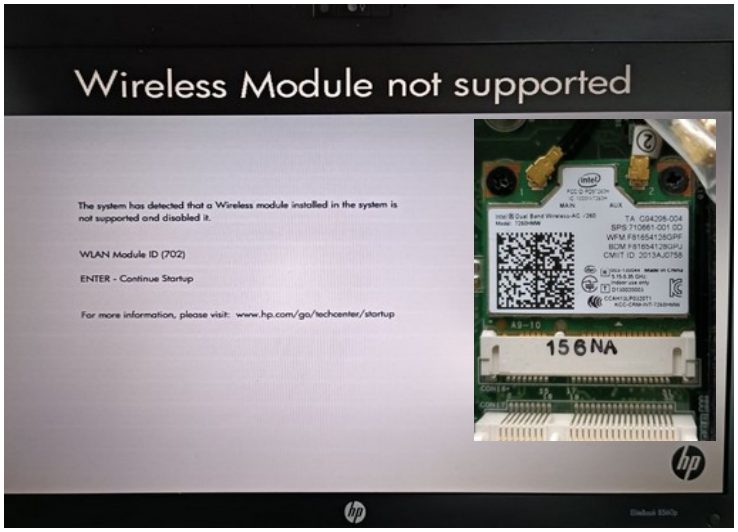
Following a recent kernel update, my Centrino WiFi card decided to destabilize, fall off the PCI bus, and take down half the kernel with it. So, I decided it was time for an upgrade.

I did my due diligence; and selected a new-old-stock Intel 7260HMW; with a genuine HP OEM part number on it; from PB-Tech. An original HP replacement part for an original HP laptop. Couldn’t get any simpler, right?

Exhibit P: Maliciously Disabling Hardware

HP Firmware – The Gift that Keeps on Giving

- ***Of course*** **HP's 2018 firmware rejects HP's own OEM-approved genuine replacement part, on sale since 2013**
- **Because, why wouldn't it?**
- **Why would anybody want a 7260HMW when they can have a Centrino 6205 with broken drivers?**



Wrong. Hewlett Packard had *seven years* to either add this module to the firmware's whitelist (which would've been reasonable, since both products were on the market at the same time), or even to just disable the whitelist when they discontinued support for the laptop (the 2018 firmware was released *long* after this model was sunset). Instead, they doubled down. They chose *violence*.

And it's one thing to show a nag screen on boot; but to power-gate the slot so the OS can't even see it? That's just *petty*. So now I have a perfectly acceptable, perfectly capable mini PCI express WiFi 5 + BT 4 card that isn't even useful as a paperweight. *Thanks, HP!*

But, throughout this journey, we haven't even answered the *most important* question yet.

Exhibit S: UEFI Booters

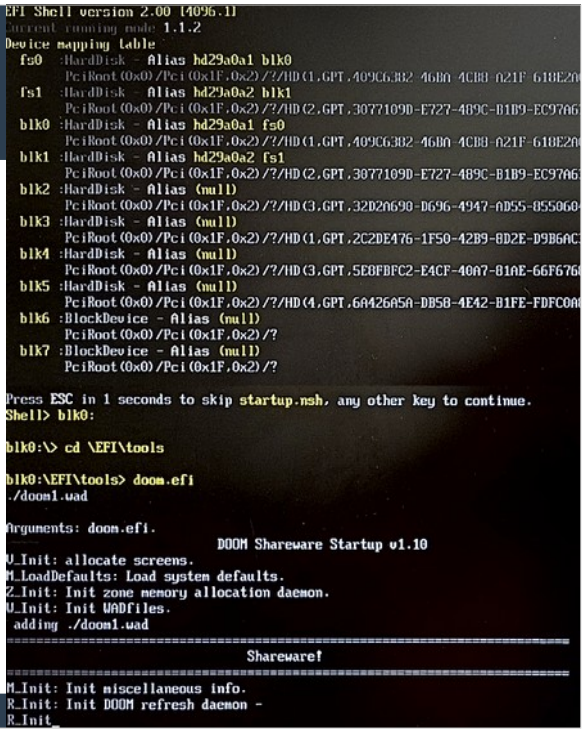
But, Can it Run DOOM?

Can it run DOOM? Well, *can it?*

Exhibit T: Subverting SMM for Frivolous Features

But, Can it Run DOOM?

- **No. Even demons tremor at the horrors of HP's mutilated Insyde BIOS**
- **HP's priority isn't booting your OS**
- **Bob wants a Bonus; so bundle in:**
  - An offline PIM EFI app (QuickLook)
  - An instant-on OS (QuickWeb, aka SplashTop)
  - An SMM calendar that replaces the Windows boot screen (DayStarter)



DOOM runs everywhere; or, at least, that's what the meme suggests.

But, you see, HP's priority was not to implement UEFI in any way that is actually useful to end users. If it can multitask a Personal Information Manager in System Management Mode while simultaneously booting Windows 7 in CSM mode; then that's "Enterprise Ready" and Bob gets his bonus. Obviously, a firmware should *not* be trusted to manage your hardware if it can't even manage your *schedule.*

DOOM died the moment it tried to draw a pixel. When prompted, the AI was unable to determine if this is a bug or a feature.

So there you have it.
After about a week of negotiation, firmware psychology, and light identity fraud…
my EliteBook finally boots like a normal computer. Well, *almost*.

HP: “I don’t wanna boot Linux.”
Me: “You’re gonna.”
HP: “NOOT NOOT!”
Me: *disguises rEFInd as Windows*
HP: “Welcome back, old friend!”

rEFInd now lives in witness protection, dressed as Windows,
and Fedora loads like nothing ever happened.

Thank you for joining me.
And remember: next time someone tells you “UEFI just works,”
NOOT NOOT.

And if there’s anybody here from HP; I’d love a quick word. *Outside!*