

Linux e il ponte USB-NVMe Realtek RTL9210

Riepilogo:

- **Sintomi:** Ripetuti reset USB, errori di I/O o dischi che scompaiono sotto Linux.
- **Interessati:** Realtek RTL9210 (confermato) e RTL9220 (possibile).
- **Causa:** Ritorno alla ROM interna (**f0.01**) dopo un errore di checksum.
- **Impatto:** Instabilità permanente, nessun strumento di reflash disponibile per Linux.
- **Soluzione:** Solo le utility Windows OEM possono ripristinare il firmware - Realtek blocca le alternative open-source.

Preambolo

Nell'anno 2025, dovrebbe essere del tutto ragionevole avviare un Raspberry Pi da un SSD collegato tramite USB. Tuttavia, a causa delle peculiarità del firmware di Realtek, questo obiettivo ragionevole è diventato un'avventura. Dopo mesi di instabilità inspiegabile - reset casuali, dischi che scompaiono, file system corrotti - l'autore ha esaurito ogni soluzione ordinaria: cavi nuovi, hub alimentati, aggiornamenti del kernel, modifiche USB e ottimizzazione del firmware. La svolta è arrivata solo quando ChatGPT ha risposto a una strana domanda a tarda notte: "È possibile che il ponte USB-NVMe sia tornato a un firmware vecchio?"

Introduzione

Se il tuo enclosure NVMe basato su Realtek diventa improvvisamente instabile dopo settimane di funzionamento impeccabile - reset USB ripetuti, errori di I/O o dischi che scompaiono - non sei solo. Questo schema è apparso in diversi marchi, da unità senza nome a OEM noti come Sabrent e Orico. Il denominatore comune: **i chip di ponte USB-NVMe Realtek RTL9210 e, possibilmente, RTL9220.**

Inizialmente, tutto funziona. Poi, apparentemente senza motivo, il dispositivo inizia a disconnettersi sotto carico o durante un uso prolungato, specialmente su sistemi Linux o Raspberry Pi. La vera causa non è l'SSD né l'alimentazione - è il controller del firmware stesso che torna silenziosamente al suo **codice di fallback incorporato nella ROM**, una versione che Realtek ancora fornisce internamente come **f0.01**.

Il meccanismo nascosto - Ritorno del firmware per progettazione

I chip di ponte Realtek memorizzano il loro firmware operativo e i dati di configurazione in un flash SPI esterno. All'avvio, il controller verifica un semplice checksum. Se quel checksum non corrisponde, rifiuta di caricare il firmware esterno e si avvia invece dalla sua ROM interna.

Questo firmware di fallback è vecchio e difettoso. Manca di numerosi aggiornamenti di stabilità USB e miglioramenti nella gestione dello stato del collegamento presenti nelle revisioni successive, portando alla classica sequenza che ogni utente Linux riconosce:

```
usb 3-2: reset del dispositivo USB ad alta velocità numero 2 usando xhci-hcd
usb 3-2: lettura del descrittore del dispositivo/64, errore -71
Avviso EXT4-fs (dispositivo sda2): errore di I/O durante la scrittura nell'inode ...
```

Il checksum può diventare non valido quando i dati di configurazione vengono riscritti - ad esempio, quando il ponte aggiorna le sue impostazioni di gestione dell'alimentazione o UAS - e il dispositivo perde alimentazione durante la scrittura. Il successivo avvio rileva un checksum corrotto e torna permanentemente al firmware della ROM.

A quel punto, il tuo "enclosure NVMe ad alte prestazioni" si comporta esattamente come il più economico guscio senza nome, perché internamente ora esegue lo stesso codice base difettoso incorporato nel silicio.

Verifica del problema

Puoi confermare questo stato facilmente sotto Linux:

```
lsusb -v | grep -A2 Realtek
```

Un ponte Realtek sano riporta una revisione del firmware (**bcdDevice**) superiore a 1.00. Uno tornato indietro mostra:

```
bcdDevice f0.01
```

Questa firma **f0.01** significa che il controller si avvia dalla ROM - e nessuna quantità di scollegamenti, riformattazioni o regolazioni del kernel lo risolverà.

Questo meccanismo di ritorno è stato **confermato sul RTL9210**. Il **RTL9220** sembra condividere la stessa architettura di design e disposizione del firmware, quindi potrebbe esibire un comportamento identico, ma ciò rimane **probabile piuttosto che provato**.

Perché non puoi ripararlo da solo

In linea di principio, la soluzione è semplice: riflashare il firmware corretto su SPI. In pratica, Realtek rende ciò impossibile.

L'azienda fornisce un aggiornamento a codice chiuso per Windows a OEM e integratori. Agli utenti Linux non viene offerto nulla. Gli sviluppatori della comunità hanno effettuato il reverse engineering di utilità di flash compatibili (**rtsupdater**, **rtl9210fw**, **rtsupdater-cli**)

che consentivano il ripristino completo del firmware da sistemi Linux - finché Realtek non ha emesso **avvisi di rimozione DMCA** per sopprimerli.

Non c'è una razionale giustificazione di proprietà intellettuale per bloccare tali utilità: non espongono microcodice, ma orchestrano solo la sequenza di aggiornamento tramite USB. Le rimozioni di Realtek non riguardavano la protezione. Erano ideologiche.

Il costo di un'ideologia

Non si tratta di idealismo open-source. Si tratta dell'**ostilità ideologica di un fornitore di hardware verso i sistemi aperti** che rompe dispositivi commercializzati come *compatibili con Linux*.

La resistenza di Realtek alla documentazione e agli strumenti aperti persiste da due decenni, coprendo Wi-Fi, Ethernet, audio e ora controller di archiviazione. Questa insularità potrebbe passare inosservata in un mondo solo Windows, ma diventa tossica quando gli stessi chip sono integrati in prodotti multi-piattaforma come il **Sabrent EC-SNVE**, che mostra apertamente il logo Linux sulla sua confezione.

Proibendo le utilità di flash per Linux e bloccando la manutenzione della comunità, Realtek ha effettivamente **criminalizzato l'autoriparazione**. Le conseguenze si propagano verso l'esterno:

- Gli utenti Linux vedono l'hardware "supportato" degradarsi in instabilità.
- Gli OEM come Sabrent e Orico affrontano costi inutili di RMA e garanzia.
- La reputazione di lunga data di Realtek per una scarsa compatibilità con Linux viene ulteriormente rafforzata.

In definitiva, non è l'open-source a rompere i dispositivi Realtek - è **l'ostilità di Realtek verso l'open-source** che li rompe.

Un percorso razionale verso il futuro

La soluzione non richiede un cambiamento ideologico, solo pragmatismo. Realtek potrebbe:

1. Rilasciare un'utilità di aggiornamento a riga di comando firmata dal fornitore per Linux (nessuna divulgazione del codice sorgente necessaria).
2. Pubblicare l'algoritmo di checksum affinché gli integratori possano validare in sicurezza le immagini flash.
3. Adottare una modalità in stile DFU che accetti aggiornamenti tramite archiviazione di massa USB, indipendentemente dal sistema operativo.

Ognuno di questi preverrebbe i costi di garanzia, proteggerebbe le relazioni con gli OEM e ripristinerebbe la fiducia nei chip di ponte Realtek tra gli utenti Linux professionali - dai costruttori di workstation agli sviluppatori Raspberry Pi.

Cosa puoi fare

Se sospetti che il tuo enclosure sia tornato al firmware ROM:

- Controlla con **lsusb -v | grep bcdDevice**.
- Se mostra **f0.01**, segnala il problema al tuo OEM.
- Includi l'estratto di **dmesg** e fai riferimento a questo meccanismo di rollback documentato.
- Chiedi al tuo fornitore di escalare il problema a Realtek, citando la necessità di un'utilità di aggiornamento compatibile con Linux.

La politica del firmware di Realtek non solo incomoda gli appassionati; crea perdite finanziarie tangibili per il proprio ecosistema. Prima questa realtà viene riconosciuta all'interno dell'azienda, prima gli utenti Linux e i partner OEM possono smettere di sprecare tempo in cicli RMA evitabili.

Risposte dei produttori

Sia Realtek che Sabrent sono stati invitati a fornire dichiarazioni riguardo al problema di rollback del firmware descritto sopra. Le loro risposte - se ricevute - saranno aggiunte qui.

Appendice - Identificazione dei dispositivi interessati

Control-ler	ID Fornitore	ID Prodotto	Note	Stato
RTL9210	0x0bda	0x9210	Ponte USB 3.1 Gen 2 10 Gb/s	Confermato comportamento di rollback
RTL9220	0x0bda	0x9220	Ponte USB 3.2 Gen 2x2 20 Gb/s	Possibile , architettura simile

Firma di rollback del firmware: **bcdDevice f0.01**

Revisioni stabili conosciute: **1.23 - 1.31**