

Linux et le pont USB vers NVMe Realtek RTL9210

Résumé :

- **Symptômes** : Réinitialisations USB répétées, erreurs d'E/S, ou disques disparaissant sous Linux.
- **Concernés** : Realtek RTL9210 (confirmé) et RTL9220 (peut-être).
- **Cause** : Retour à la ROM interne (**f0.01**) après un échec de la somme de contrôle.
- **Impact** : Instabilité permanente, aucun outil de reflashage disponible pour Linux.
- **Solution** : Seuls les utilitaires Windows OEM peuvent restaurer le firmware - Realtek bloque les alternatives open-source.

Préambule

En 2025, il devrait être tout à fait raisonnable de démarrer un Raspberry Pi à partir d'un SSD connecté via USB. Pourtant, en raison des particularités du firmware de Realtek, cet objectif raisonnable est devenu une aventure. Après des mois d'instabilité inexplicable – réinitialisations aléatoires, disques disparaissant, systèmes de fichiers corrompus – l'auteur a épuisé toutes les solutions habituelles : nouveaux câbles, hubs alimentés, mises à jour du noyau, ajustements USB, et réglages du firmware. La percée n'est survenue que lorsque ChatGPT a répondu à une question étrange posée tard dans la nuit : « Est-il possible que le pont USB vers NVMe soit revenu à un ancien firmware ? »

Introduction

Si votre boîtier NVMe basé sur Realtek devient soudainement instable après des semaines de fonctionnement sans faille – réinitialisations USB répétées, erreurs d'E/S, ou disques disparaissant – vous n'êtes pas seul. Ce schéma est apparu chez plusieurs marques, des unités sans nom aux OEM bien connus comme Sabrent et Orico. Le dénominateur commun : **les puces de pont USB vers NVMe Realtek RTL9210 et, peut-être, RTL9220.**

Au début, tout fonctionne. Puis, apparemment sans raison, l'appareil commence à se déconnecter sous charge ou lors d'une utilisation prolongée, en particulier sur les systèmes Linux ou Raspberry Pi. La véritable cause n'est ni le SSD ni l'alimentation – c'est le contrôleur de firmware lui-même qui revient discrètement à son **code de secours intégré dans la ROM**, une version que Realtek continue d'expédier en interne sous le nom **f0.01**.

Le mécanisme caché – Retour du firmware par conception

Les puces de pont Realtek stockent leur firmware opérationnel et leurs données de configuration dans une mémoire flash SPI externe. Au démarrage, le contrôleur vérifie une simple somme de contrôle. Si cette somme ne correspond pas, il refuse de charger le firmware externe et démarre à la place depuis sa ROM interne.

Ce firmware de secours est ancien et défectueux. Il manque plusieurs correctifs de stabilité USB et améliorations de la gestion de l'état de la liaison présents dans les révisions ultérieures, ce qui conduit à la séquence classique que tout utilisateur Linux reconnaît :

```
usb 3-2 : réinitialisation du dispositif USB à haute vitesse numéro 2 utilisant
xhci-hcd
usb 3-2 : lecture du descripteur de dispositif/64, erreur -71
Avertissement EXT4-fs (dispositif sda2) : erreur d'E/S lors de l'écriture dans l'in-
ode ...
```

La somme de contrôle peut devenir invalide lorsque les données de configuration sont ré-écrites – par exemple, lorsque le pont met à jour ses paramètres de gestion d'énergie ou UAS – et que l'appareil perd l'alimentation en cours d'écriture. Le démarrage suivant détecte une somme de contrôle corrompue et revient de manière permanente au firmware de la ROM.

À ce stade, votre « boîtier NVMe haute performance » se comporte exactement comme le boîtier sans nom le moins cher, car il exécute désormais en interne le même code de base défectueux gravé dans le silicium.

Vérification du problème

Vous pouvez confirmer cet état facilement sous Linux :

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

Un pont Realtek sain rapporte une révision de firmware (**bcdDevice**) supérieure à 1.00. Un pont revenu en arrière affiche :

```
bcdDevice f0.01
```

Cette signature **f0.01** signifie que le contrôleur démarre depuis la ROM – et aucun débranchement, reformatage ou réglage du noyau ne le corrigera.

Ce mécanisme de retour en arrière a été **confirmé sur le RTL9210**. Le **RTL9220** semble partager la même architecture de conception et la même disposition du firmware, il pourrait donc présenter un comportement identique, mais cela reste **probable plutôt que prouvé**.

Pourquoi vous ne pouvez pas le réparer vous-même

En principe, la solution est simple : reflasher le firmware correct sur le SPI. En pratique, Realtek rend cela impossible.

L'entreprise fournit un utilitaire de mise à jour à code fermé pour Windows aux OEM et intégrateurs. Rien n'est offert aux utilisateurs Linux. Les développeurs communautaires ont effectué une ingénierie inverse sur des utilitaires de flash compatibles (**rtsupdater**, **rtl9210fw**, **rtsupdater-cli**) qui permettaient une restauration complète du firmware depuis les systèmes Linux – jusqu'à ce que Realtek émette des **notifications de retrait DMCA** pour les supprimer.

Il n'y a aucune justification plausible en matière de propriété intellectuelle pour bloquer de tels utilitaires : ils n'exposent pas de microcode, ils orchestrent simplement la séquence de mise à jour via USB. Les retraits de Realtek n'étaient pas pour la protection. Ils étaient idéologiques.

Le coût d'une idéologie

Il ne s'agit pas d'idéalisme open-source. Il s'agit de l'**hostilité idéologique d'un fabricant de matériel envers les systèmes ouverts** qui casse des appareils commercialisés comme *compatibles Linux*.

La résistance de Realtek à la documentation et aux outils ouverts persiste depuis deux décennies, couvrant le Wi-Fi, l'Ethernet, l'audio, et maintenant les contrôleurs de stockage. Cette insularité pourrait passer inaperçue dans un monde uniquement Windows, mais elle devient toxique lorsque ces mêmes puces sont intégrées dans des produits multi-plateformes comme le **Sabrent EC-SNVE**, qui affiche ouvertement le logo Linux sur son emballage.

En interdisant les utilitaires de flashage Linux et en bloquant la maintenance communautaire, Realtek a effectivement **criminalisé l'auto-réparation**. Les conséquences se propagent vers l'extérieur :

- Les utilisateurs Linux voient le matériel « pris en charge » dégénérer en instabilité.
- Les OEM comme Sabrent et Orico font face à des coûts inutiles de RMA et de garantie.
- La réputation de longue date de Realtek pour une mauvaise compatibilité Linux est à nouveau renforcée.

En fin de compte, ce n'est pas l'open-source qui casse les appareils Realtek – c'est l'**hostilité de Realtek envers l'open-source** qui les casse.

Une voie rationnelle vers l'avant

La solution ne nécessite aucun changement idéologique, seulement du pragmatisme. Realtek pourrait :

1. Publier un utilitaire de mise à jour en ligne de commande signé par le fabricant pour Linux (aucune divulgation de code source nécessaire).
2. Publier l'algorithme de somme de contrôle pour que les intégrateurs puissent valider les images flash en toute sécurité.

3. Adopter un mode de type DFU qui accepte les mises à jour via le stockage de masse USB, indépendamment du système d'exploitation.

Chacun de ces éléments éviterait les coûts de garantie, protégerait les relations avec les OEM et restaurerait la confiance dans les puces de pont Realtek parmi les utilisateurs professionnels de Linux – des constructeurs de stations de travail aux développeurs Raspberry Pi.

Ce que vous pouvez faire

Si vous pensez que votre boîtier est revenu au firmware ROM :

- Vérifiez avec **lsusb -v | grep bcdDevice**.
- S'il affiche **f0.01**, signalez le problème à votre OEM.
- Incluez l'extrait de **dmesg** et mentionnez ce mécanisme de retour documenté.
- Demandez à votre fournisseur d'escalader le problème auprès de Realtek, en citant le besoin d'un utilitaire de mise à jour compatible Linux.

La politique de firmware de Realtek ne gêne pas seulement les passionnés ; elle crée des pertes financières tangibles pour son propre écosystème. Plus tôt cette réalité sera reconnue au sein de l'entreprise, plus tôt les utilisateurs Linux et les partenaires OEM pourront cesser de perdre du temps dans des cycles RMA évitables.

Réponses des fabricants

Realtek et Sabrent ont tous deux été invités à fournir des déclarations concernant le problème de retour du firmware décrit ci-dessus. Leurs réponses – si reçues – seront ajoutées ici.

Annexe – Identification des appareils concernés

Contrôleur	ID Fournisseur	ID Produit	Notes	Statut
RTL9210	0x0bda	0x9210	Pont USB 3.1 Gen 2 10 Gb/s	Confirmé comportement de retour
RTL9220	0x0bda	0x9220	Pont USB 3.2 Gen 2x2 20 Gb/s	Possible , architecture similaire

Signature de retour du firmware : **bcdDevice f0.01**

Révisions stables connues : **1.23 – 1.31**