

MARCHANDS, PRESTATAIRES ET ISO

PHP2Shell : ce que les nouvelles failles de PHP signifient pour quiconque accepte des paiements par carte

Publié le 22 octobre 2026

Deux failles critiques de PHP peuvent permettre à un attaquant non authentifié d'exécuter du code sur un serveur web. Pour un marchand, c'est un risque pour l'intégrité de la page de paiement, les identifiants de paiement et la disponibilité. Pour un prestataire, c'est un risque de portefeuille. Ce qui s'est passé, ce qu'il faut vérifier et ce qu'il faut demander à votre hébergeur.

Le 21 octobre 2026, le projet PHP a publié des correctifs d'urgence pour deux failles critiques du langage lui-même, référencées CVE-2026-109618 et CVE-2026-109619 et surnommées **PHP2Shell**. Toutes deux sont notées 9,2 sur l'échelle CVSS 4.0. L'une comme l'autre peut permettre à un attaquant non authentifié d'exécuter du code sur un serveur web concerné, et les failles se trouvent dans PHP, pas dans une application particulière bâtie dessus.

Pour les marchands, cela crée un risque pour l'intégrité de la page de paiement, les identifiants de paiement et la disponibilité. Pour les prestataires de paiement et les ISO, cela crée une exposition possible sur l'ensemble de leur portefeuille de marchands. Voici ce qui s'est passé, ce qu'il faut vérifier et ce qu'il faut demander à votre hébergeur.

PHP fait tourner une très grande part du web, y compris la plupart des logiciels de boutique qu'utilisent les petits et moyens marchands. Cela fait de cette affaire une histoire de paiement autant que de sécurité.

Ce qui a été découvert

- **CVE-2026-109618** est un bogue de corruption de mémoire. Si PHP manque de mémoire en pleine préparation d'une requête, le nettoyage qui suit peut traiter comme valides des données à moitié construites.
- **CVE-2026-109619** est un bogue d'état obsolète, de type use-after-free. PHP peut conserver un état interne qui pointe vers de la mémoire de requête déjà libérée, et un traitement ultérieur peut

l'utiliser. PHP cite comme effets l'exécution de code à distance et la divulgation de mémoire sensible.

- **Portée.** Le code concerné le plus ancien remonte à PHP 5.4. Le projet PHP indique que toute installation qui sert des requêtes web est exposée, qu'elle fonctionne avec PHP-FPM, Apache avec mod_php, CGI, FrankenPHP ou LiteSpeed. PHP juge le risque le plus élevé avec mod_php, où l'attaque démontrée peut aboutir avant même l'exécution du moindre code du site. Cela décrit l'attaque qui a été montrée, pas une affirmation que les autres configurations seraient sûres.
- **Versions corrigées.** PHP 8.5.12, 8.4.27, 8.3.36 et 8.2.35. Les versions 7.4 à 8.1 n'ont pas de version officielle, seulement des rétroportages minimaux du code source. Les versions 5.4 à 7.3 sont en fin de vie et ne seront pas corrigées du tout par le projet PHP. Certaines distributions Linux et certains éditeurs de support à long terme rétroportent bien les correctifs vers des branches plus anciennes : consultez donc l'avis de sécurité de votre propre fournisseur avant de supposer qu'une installation donnée n'est pas corrigée.

Où en sont les choses

Les détails ont fuité le 20 octobre, un jour avant la publication prévue, qui avait elle-même été avancée depuis le 22 octobre. L'équipe de renseignement sur les menaces d'Amazon a signalé une tentative contre ses honeypots le 17 octobre, quatre jours avant la publication, et d'autres depuis, apparemment de plusieurs acteurs. Cloudflare et watchTowr signalent du trafic d'attaque, et watchTowr affirme avoir reproduit les deux failles. Un code d'exploitation public existe, même si sa fiabilité n'est pas vérifiée, et le projet PHP a déclaré lors de la publication ne pas avoir identifié d'exploit fonctionnel. On ne sait pas encore si les tentatives observées jusqu'ici réussiraient contre de vrais systèmes en production.

C'est une menace crédible et urgente. Des tentatives d'exploitation et du code de démonstration ne prouvent pas, à eux seuls, que des données de paiement ont été volées à des marchands, et au 22 octobre nous n'avons connaissance d'aucune compromission confirmée de données de paiement attribuable à PHP2Shell. Corrigez dès maintenant les systèmes concernés, enquêtez sur tout ce qui paraît suspect et conservez les journaux utiles. L'absence de violation confirmée n'est pas une raison d'attendre.

Pourquoi cela compte pour les paiements par carte

Un marchand n'a pas besoin d'exécuter du code PHP qui lui est propre pour être touché. Le logiciel sous la boutique suffit, et les conséquences atteignent le paiement de quatre façons.

1. **Le serveur qui construit la page de paiement.** L'exécution de code sur ce serveur permet à un attaquant de modifier ce que fait la page de paiement. Y injecter un script qui copie les données de carte pendant que les clients les saisissent (souvent appelé skimming ou attaque de type

Magecart) est la voie classique par laquelle des données de carte volées quittent les boutiques PHP.

2. **Identifiants et clés.** Les clés d'API de la passerelle de paiement, les secrets de webhook et les accès d'administration se trouvent généralement dans la configuration de la boutique. Quiconque peut exécuter du code sur le serveur peut les lire.
3. **Fraude et abus.** Des clés volées peuvent servir à émettre des remboursements ou à lancer des transactions de test. Un site compromis peut aussi héberger une seconde page de paiement cachée, qui ressemble beaucoup, pour le prestataire, à du [transaction laundering](#).
4. **Indisponibilité.** Même une attaque qui échoue peut faire planter des processus de travail. Une boutique à l'arrêt ne prend pas de commandes, et une remise en route qui dure plusieurs jours peut perturber les cycles de [facturation des abonnements](#) et provoquer remboursements et litiges.

Les marchands qui redirigent vers la page de paiement hébergée du prestataire, ou qui intègrent les champs du prestataire lui-même, sont mieux placés : le numéro de carte ne se trouve jamais sur leur serveur. Ils restent toutefois exposés sur l'administration, les remboursements et les clés d'API décrits ci-dessus.

Ce qu'il faut vérifier cette semaine

- **Découvrez quel PHP vous utilisez.** Sur votre propre serveur, exécutez `php -v`. Sur un hébergement géré ou mutualisé, consultez le panneau de contrôle ou interrogez l'hébergeur. Les extensions et les logiciels de boutique ne l'indiquent pas, et une boutique peut rester sur une ancienne version de PHP des années après la mise à jour de son logiciel.
- **Appliquez le correctif de l'éditeur.** Pour la plupart des distributions Linux et des hébergeurs, cela signifie la mise à jour de paquet du fournisseur, suivie d'un redémarrage de PHP pour que le changement prenne effet. Ne vous fiez pas au seul numéro de version de PHP, car les distributions rétroportent souvent le correctif sans le modifier.
- **Si vous êtes sur PHP 7.3 ou antérieur, prévoyez de migrer.** Le projet PHP ne corrigera pas ces versions. Demandez à votre hébergeur ou à votre développeur si leur fournisseur rétroporte un correctif, et demandez la date, pas l'intention.
- **Les mesures d'atténuation sont temporaires.** Le projet PHP documente des réglages qui limitent chaque faille, et un pare-feu applicatif web ou des restrictions réseau peuvent réduire l'exposition pendant que vous préparez une mise à jour. Aucune n'est un correctif complet. Passez à une version corrigée ou appliquez dès que possible le correctif de sécurité vérifié de votre distribution, et si un système exposé à internet ne peut pas être corrigé rapidement, envisagez de restreindre son exposition.
- **Changez les secrets si vous ne pouvez pas exclure une compromission.** Clés de la passerelle de paiement, secrets de webhook, mots de passe d'administration et identifiants de base de

données. Recherchez de nouveaux comptes administrateur, des fichiers de paiement modifiés et des scripts inattendus sur la page de paiement.

- **Examinez la page de paiement.** Comparez les scripts qu'elle charge avec ce que vous attendez. Des domaines tiers inconnus sont le signal d'alerte.

PCI DSS et votre prestataire

Si vous relevez du périmètre PCI DSS, cela entre dans vos responsabilités existantes de gestion des vulnérabilités. L'exigence 6.3.1 impose d'identifier les nouvelles vulnérabilités et de leur attribuer une cote de risque. L'exigence 6.3.3 impose que les correctifs critiques ou de haute sécurité, selon cette cote, soient installés dans le mois qui suit leur publication.

C'est un maximum, pas une recommandation d'attendre. Vu la gravité de PHP2Shell et les signalements d'activité d'exploitation, les systèmes concernés exposés à internet doivent être corrigés immédiatement. Lorsque la page de paiement est dans le périmètre, les exigences relatives aux scripts des pages de paiement peuvent aussi être pertinentes.

Un marchand qui ne peut pas démontrer qu'il a corrigé rapidement, et qui n'a aucune trace du motif, se trouve en position fragile si quelque chose tourne mal. Une courte note écrite indiquant quand la faille a été identifiée, ce qui a été fait et quand suffit pour commencer.

Prévenez tôt votre prestataire si vous avez des raisons de penser que la boutique a été compromise. Les prestataires de paiement peuvent avoir des obligations de signalement et d'escalade au titre des règles applicables des réseaux de cartes, et un marchand qui le soulève dès le premier jour est traité très différemment de celui que le prestataire découvre par une alerte du réseau ou une série de signalements de fraude.

Pour les prestataires et les ISO

L'exposition touche tout votre portefeuille et n'est pas répartie uniformément. Les marchands sur un PHP en fin de vie, sur un hébergement mutualisé bon marché et sur un logiciel de boutique géré par eux-mêmes sont les premiers à examiner. Des mesures raisonnables sont :

- Contacter les marchands sur un logiciel de boutique auto-hébergé avec un avis court et clair, en leur demandant la date à laquelle ils ont appliqué le correctif.
- Surveiller les signes d'une page de paiement compromise : hausse soudaine de tests de cartes, refus inattendus, remboursements qui ne correspondent pas aux commandes, ou litiges signalant une fraude sur des cartes utilisées uniquement chez ce marchand.
- Relancer la surveillance des sites web sur les marchands au plus gros volume et rechercher de nouveaux scripts ou des destinations de paiement inconnues.
- Se rappeler que la même faille touche vos propres applications et les sites de vos partenaires et affiliés.

À l'enrôlement, une question utile est de savoir quelle plateforme et quel hébergement le marchand utilise, et qui est chargé de les maintenir à jour. La réponse en dit long sur la façon dont un compte se comportera lors d'une semaine comme celle-ci.

Sources

- [Tenable: PHP2Shell \(CVE-2026-109618, CVE-2026-109619\) frequently asked questions](#)
- [Qualys: PHP announces critical security release for two vulnerabilities](#)
- [Cisco Talos: Two critical RCEs in PHP core](#)
- [CERT-EU: Critical vulnerabilities in PHP \(2026-018\)](#)
- [watchTowr: PHP remote code execution exploited](#)

Lire en ligne : <https://payments-marketplace.com/fr-FR/insights/php2shell-payments/>

Payments Marketplace Ltd

Registered in England & Wales No. 17424288

Garden Ctg, Dereham Road, Dereham, Norfolk, NR19 2PW, UK

connect@payments-marketplace.com · +44 1366 727140