



How to Revoke Every WordPress Application Password After a Hack

A WordPress password reset never revokes Application Passwords.
How to remove every one after a hack, one profile at a time or in one
click with mySites.guru.

Phil E. Taylor | 29 September 2026

You clean up a hacked WordPress site the right way. Remove the malware, patch the plugin that let it in, change every password, force everyone to log back in. Weeks later something is still pulling data through the REST API, using a credential no one remembers issuing.

Application Passwords are not your login password. They are a separate credential, stored in their own place, and a password reset does not touch them. One created during a compromise keeps authenticating REST API requests exactly as before, right through a cleanup that changed every password on the site and never looked at it.

Revoke Every Application Password From mySites.guru in One Click

mySites.guru's [Revoke Every Application Password After a Hack](#) tool clears every Application Password on a site, across every user, in a single action. It is a manual tool you run when you need it rather than a background check, reached from the tool finder (Cmd-K, or Ctrl-K on Windows) or the link on a hacked site's card.

Revoke every password

Deletes every user's Application Passwords on the site in one action, not one profile at a time.

The core method runs first

It calls WordPress's own `delete_all_application_passwords()` for each user, so the standard `wp_delete_application_password` action still fires and any security plugin still logs it.

Then it checks the raw data

It re-reads each user's usermeta afterwards. If a filter on a compromised site blocked the delete without saying so, it writes an empty value directly and tells you it had to.

Log everyone out

A separate, optional step that ends every session token as well. Off by default, because it also ends your own session if you reached the site through mySites.guru.

That second check matters more here than it would on a clean site. A site that is actively compromised can have a malicious filter on `update_user_metadata` that reports success while writing nothing, which is exactly the failure mode a normal admin-screen click cannot see. Calling the real WordPress function first means the usual hooks and logging still fire; reading the raw row afterwards means a lie from a compromised filter does not end the story.

The same double-check runs for the logout step: WordPress's own

`WP_Session_Tokens::destroy_all()` for each user, a raw re-count, and a direct delete of anything still sitting in the sessions table.

What Are WordPress Application Passwords?

Application Passwords are per-user credentials for the WordPress REST API, XML-RPC and WP-CLI, authenticated over standard HTTP Basic Auth without exposing the account's real login password. WordPress core added them in version 5.6, [announced on Make WordPress Core in November 2020](#) and released that December.

Each one is stored under `_application_passwords` in that user's usermeta, as a bcrypt hash alongside a name, a created date, a last-used timestamp and the last IP address that used it, [per WordPress's own documentation](#). WordPress hashes the password the same way it hashes a login password, but that is where the resemblance to your login

credential ends: the two are unrelated values, generated independently, and changing one has no effect on the other.

Why a Password Reset Does Not Touch Them

A password change updates the `user_pass` column. An Application Password lives entirely in `usermeta` and is never read or written when `user_pass` changes, so the two operations simply never meet. [Recovery guidance from TrustedLogin](#) makes the same point: reviewing and revoking Application Passwords is a distinct step from a password reset, not a side effect of one.

Most general hack-recovery checklists still lead with “change every password” and leave it there, because that instruction covers the credential people think about first. Application Passwords were built specifically so that automated tools, mobile apps and integrations would not need the account’s real password at all, which is exactly why revoking the real password leaves them completely unaffected.

Those same checklists also stop at the technical fix, every one we have found. Whether anyone outside your own team now has to be told what an Application Password could reach is a separate question, and it gets its own section near the end of this post, not because it matters least.

How Do You Revoke Application Passwords by Hand?

WordPress core already has a revoke button, and it is worth knowing exactly what it covers before assuming it solves this for a whole site:

1. In wp-admin, go to **Users**, then **All Users**, and click **Edit** on the account you want to check.
2. Scroll to the **Application Passwords** section of that user’s profile.
3. Click **Revoke** next to a single password to remove just that one, or **Revoke All Application Passwords** at the bottom of the list to clear every password that

user holds.

4. Repeat for every user on the site. The button only ever acts on the profile you are currently viewing, so there is no way to reach every user's passwords from one screen.
5. Confirm from outside with the old credential:

```
curl -u "username:revoked-app-password" https://example.com/wp-json/wp/v2/
```

A revoked password answers 401. If you manage sites from the command line, WP-CLI has a per-user equivalent, `wp user application-password delete <id> --all`, which is faster to run in a loop than clicking through wp-admin but is still a per-user command, not a site-wide one.

Core's Revoke All button is real, and it is scoped to one profile

It is not a missing feature so much as a design choice: Application Passwords live under each user's own profile because that is where a person manages their own credentials. Nothing in WordPress core was ever built to act on every user's passwords from a single screen, so a site-wide clean-up step means visiting every profile in turn.

Is There a Site-Wide Revoke in Any Security Plugin?

Not one that revokes every user's existing Application Passwords in a single click, as far as we have found. Wordfence can disable Application Passwords entirely, which blocks new ones and stops existing ones from authenticating, but that is a different action from clearing them out, and it stays in effect as a standing restriction rather than a one-time cleanup step. If a plugin you use does offer a real site-wide revoke, treat that as a reason to check it works the way this post describes: calling WordPress's real deletion function first, then verifying the raw data, not just clearing a list on screen.

What About WooCommerce REST API Keys?

Left alone, on purpose. WooCommerce API keys are a separate credential system with their own storage and their own revoke screen under WooCommerce's settings, distinct from core Application Passwords. mySites.guru's tool does not touch them, so if a hacked site also runs a WooCommerce store with API keys issued to an integration, revoke those separately, one at a time, from WooCommerce's own API keys list.

Does Revoking Application Passwords Stop the Site Being Hacked Again?

No. It closes one specific door, a REST API credential that a password reset and a file cleanup both leave standing. It does not remove malware, patch the plugin that let someone in, or clean up an account an attacker planted. Pair it with [cleaning up a hacked WordPress or Joomla site](#) and our guide to [checking whether your WordPress site is actually hacked](#) if you are still confirming what happened. If you would rather hand the whole thing to someone, [fix.mysites.guru](#) is a single fixed fee of £120 per incident, usually same day, and non-subscribers get a free month included.

The Step After the Technical Cleanup: Do You Need to Report a Breach?

Every checklist covered so far, ours included until this post, stops at the technical fix: remove the malware, patch the plugin, revoke the credentials. None of it asks the other question a hack can raise, which is whether someone outside your team now has to be told.

If there is a reasonable chance an attacker used a live Application Password to read personal data through the REST API, customer names, email addresses, order records from a WooCommerce store, anything that identifies a real person, that is very likely a personal data breach under UK GDPR, or the equivalent under the EU GDPR for an EU-established controller, or your own jurisdiction's data protection law elsewhere. The law does not care that the credential looked legitimate. An Application Password pulling the

orders endpoint unnoticed for a month and a defaced homepage are both a breach if personal data was exposed, and only the second one looks like an incident.

Two duties then run on their own clock, alongside the technical cleanup rather than after it:

- Tell your supervisory authority. In the UK that is the Information Commissioner's Office, and the rule is 72 hours from when you become aware of the breach, not from when the password was first created, unless the breach is unlikely to put anyone's rights and freedoms at risk.
- Tell the people affected, without undue delay, if the breach is likely to result in a high risk to them, for example because what leaked could be used for fraud or identity theft.

Working out whether either duty applies takes evidence the cleanup already produces: when the password was created, what it could reach through the REST API, and how long it went unnoticed. That is one more reason to revoke it and write down what you found in the same sitting, rather than treating the technical fix as the end of the incident. This is not legal advice, and the thresholds differ by jurisdiction, so involve whoever handles data protection for the business, or a solicitor, rather than deciding it alone from a blog post.

Where mySites.guru Fits

Checking every user's Application Passwords by hand does not scale past one or two sites, which is exactly when it stops happening. A mySites.guru subscription covers the rest of a hack response from one dashboard:

- The WordPress checks covered in five new checks for WordPress hacks a file scan cannot see find rogue and hidden administrator accounts on every connected site automatically.
- Installed plugins and themes are matched against known vulnerabilities, so the entry point that led to the compromise gets flagged before it happens again.

- [Backups run before any mass change](#), so revoking credentials or updating a whole portfolio at once is never a one-way action.
- Every connected site gets the same post-hack tools, so a response that would mean visiting every user's profile on every site becomes one action per site instead.

[See the plans](#), or run a [free audit](#) on one site to see what it finds.

Further Reading

- [Application Passwords: Integration Guide, Make WordPress Core](#) - the original announcement ahead of WordPress 5.6.
- [Application Passwords, WordPress Advanced Administration Handbook](#) - the official reference for how they work and where they are stored.
- [WP Application Passwords::delete_all_application_passwords\(\)](#), [WordPress Developer Reference](#) - the core method behind the per-user Revoke All button.
- [How to revoke admin access from someone who no longer needs it, TrustedLogin](#) - on Application Passwords staying active after a password change.

Frequently Asked Questions

Does a WordPress password reset revoke Application Passwords?

No. Application Passwords are a separate credential from your login password, stored in their own usermeta entry, and changing your password does not touch them. An Application Password created before a password reset keeps authenticating the REST API exactly as before.

Can I revoke all Application Passwords for every user in one click?

Not from WordPress core's own admin screens. Each user's profile has a Revoke All Application Passwords button, but it only clears that one user's passwords, so a site with forty users needs forty visits. mySites.guru's tool does every user on the site in a single action.

What are WordPress Application Passwords for?

They authenticate the WordPress REST API (and XML-RPC and WP-CLI) over HTTP Basic Auth, without exposing the account's real login password. WordPress core added them in version 5.6, released in December 2020.

Does revoking Application Passwords log the user out of wp-admin?

No, those are two different things. Revoking Application Passwords only affects REST API and XML-RPC access. mySites.guru's tool has a separate, optional log-everyone-out step for ending wp-admin sessions, off by default because it also ends your own session if you reached the site through mySites.guru.

Are WooCommerce REST API keys covered by this?

No. WooCommerce API keys are a distinct credential system, stored separately from core Application Passwords, and this tool leaves them alone. They need revoking on their own, one at a time, from WooCommerce's own API keys screen.

Is there a Joomla equivalent of this tool?

Yes. Joomla's equivalent credential is the API Token, and mySites.guru has a matching post-hack tool that revokes every one on a connected Joomla site in one click. See revoke every Joomla API token after a hack.

Why re-check application passwords after cleaning the files?

Because a file cleanup and a plugin update do not read usermeta. Nothing about restoring core files or removing a webshell touches the `_application_passwords` entry sitting in the database, so a password created during the compromise keeps working through a cleanup that never looked at it.

Do I need to report a WordPress Application Password hack to a data protection authority?

If there is a reasonable chance the password was used to read personal data through the REST API, such as customer names, email addresses or order records, it may be a notifiable personal data breach under UK GDPR or the equivalent law in your jurisdiction, regardless of whether the credential or a webshell was how the data was read. In the UK, the ICO's rule is 72 hours from when you become aware, and affected individuals must be told without undue delay if the risk to them is high. This is not legal advice; get a definite answer from whoever handles data protection for the business.

AI MODIFIED

Written and edited by a human, with AI assistance. [Our approach to AI](#)

Get Your Free Site Audit

See how your WordPress and Joomla sites measure up.
No credit card required.

<https://manage.mysites.guru/en/register>

Get in touch

Phil E. Taylor
phil@phil-taylor.com

