mySites.guru

UP 6.1.0 fixes file read and code execution in the UP plugin for Joomla

Before UP 6.1.0, one anonymous request could read a Joomla site's configuration.php through the UP plugin, and any author could run PHP. We found it.

Phil E. Taylor | 26 September 2026

UP, the Universal Plugin, is a Joomla content plugin that reads `{up ...}` shortcodes out of article text and acts on them while the page renders. About 160 actions ship with it. Most are harmless: embed a video or a map, make a table responsive, list articles by tag. A handful do far more, running PHP, querying the database, reading files, and fetching code from GitHub. Its maintainer released UP 6.1.0 on 26 September 2026, and it fixes a set of flaws we found in 6.0.29 and reported privately in August.

The worst of them needed nothing but a web browser. UP answers requests on Joomla's anonymous AJAX endpoint, and in 6.0.29 one of those requests returned any file the server could read, `configuration.php` and the database password inside it included.

> **Update UP now**
>
> UP 6.1.0 (Joomla 5.2 to 6.x) and UP 5.2.1 (Joomla 3.10 to 5.1) are the fixed releases. The maintainer lists every version from 5.0 to 6.0.29 as affected. This post describes the mechanisms and leaves the working requests out.

TL;DR

- Update UP to **6.1.0** on Joomla 5.2 to 6.x, or **5.2.1** on Joomla 3.10 to 5.1. Every version from 5.0 to 6.0.29 is affected
- In 6.0.29 an anonymous request could make the site **download action code from GitHub** with certificate checks switched off, the route to running someone else's code
- A second anonymous request **read any file** the web server could read, `configuration.php` included
- Anyone who could **write an article** could run PHP and query the whole database, and it ran for every visitor to that page

- A third anonymous request could **create files outside the folder** they were meant for
- Tracked as CVE-2026-97160 **to** CVE-2026-97163, scored up to 10.0 Critical by the Joomla CNA
- It took fifteen release candidates over four weeks to close every route, and we retested each one, because more than once a changelog said “fixed” over code that was not
- mySites.guru flags every connected site on an affected version, so you do not have to check each one by hand

The findings, by who could exploit them

Whether a flaw needs a login decides how urgent it is, so here they are grouped that way. Three needed no login at all.

#	Flaw in 6.0.29	Who can exploit it	What it exposes	CVE
1	Anonymous trigger to install action code from GitHub	Anyone, no login	Code execution, if the download can be steered	CVE-2026-97163
2	Arbitrary file read on the AJAX endpoint	Anyone, no login	`configuration.php`, and any other file the server can read	CVE-2026-97161
3	PHP and SQL actions open to article authors	Anyone who can write an article	PHP execution and the whole database, for every visitor	CVE-2026-97160, CVE-2026-97162
4	File creation outside the intended folder	Anyone, no login	Stray files written where they should not be	CVE-2026-97161

Our original report listed a fourth anonymous flaw, a file move and delete endpoint. When the maintainer replied, he pointed out that the file had a PHP syntax error and

never ran, which we confirmed. It is tidied up in 6.1.0, but it was never exploitable, so it is not in the table.

1. An anonymous request could install code from GitHub

UP has a “mini” mode. Rather than ship all its actions, it downloads the one a page needs from the maintainer’s GitHub repository the first time it is used, and unpacks it into the plugin’s own folder, where Joomla runs it.

In 6.0.29 an anonymous visitor could set that download in motion, and the fetch ran with its TLS certificate checks switched off. On its own the request only chose which file name to ask for inside a fixed repository, so it is not a one-shot takeover. But with the certificate checks off, anyone positioned between the server and GitHub could answer with a different archive, and whatever arrived was unpacked into a folder of PHP that UP includes on demand. Our own provisional score was lower because of that second condition; the Joomla CNA scored it 10.0 Critical. We proved the anonymous trigger on our test site and stopped there, because the next step would have meant serving a payload through the maintainer’s own infrastructure. 6.1.0 restricts the download and turns the certificate checks back on.

2. Any file on the server, for anyone who asked

One action existed to show a text or CSV file in a popup. The file name came straight from the request, and the endpoint checked neither who was asking nor where the file sat on disk. The file-type limits lived in the code that draws the button, and never in the code that answered the request.

On our test site, one anonymous request returned a canary file we had placed outside the web root, and `configuration.php` came back the same way. That file holds the database username and password, so reading it is close to handing over the database. 6.1.0 restricts this action to the file types it was meant for and keeps it inside the site.

3. What an article author could reach, and how the gate kept failing

The most drawn-out part of this was not an anonymous flaw. It was the boundary between an ordinary content author and the actions that run code.

UP keeps a list of action names reserved for administrators, and when an article is saved it scans the text and blocks the save if a non-administrator used one. That check read the action name exactly as it was typed. The renderer did not. Before deciding which action to run, it rewrote the name several times: stripping editor markup, changing the case, swapping punctuation, and applying a dictionary of alternative names the plugin accepts for convenience.

Each rewrite was a way in. Typed in a form the check did not recognise but the renderer did, a shortcode saved cleanly and still ran. The clearest demonstration was a matched pair: one spelling of an action blocked, another spelling of the same action, by the same user on the same build, saved and ran. Behind that gate sat an action that runs PHP and an action that reads the database directly, so a low-privilege author could reach code execution and the users table on a page any anonymous visitor could then open.

Closing this took several tries. One release worked out the real name once, in a single function used by both the save check and the renderer, which shut all four routes at once. The next release dropped one lookup from that function and reopened one of them, which we proved live and reported the same morning; the release after restored it. The version that shipped reserves the risky actions to administrators, checks at save time and again at render time, and matches the name the same way in both places.

The PHP and SQL actions still exist for administrators, by design. So on a site running UP, an administrator writing an article can read almost any table in the database. Keep that in mind when you decide who holds administrator access. It is also a reason to be wary of any flaw elsewhere that could forge an administrator’s request.

A permission check that took the site down

One release added a missing permission check to an AJAX handler. It was the right check, written in a way that could not run: it referred to the current object inside a static function, which PHP does not allow. The moment an unauthorised visitor triggered it, PHP raised a fatal error.

On the increasingly common FrankenPHP setup that was not a tidy error page. The failure produced a response the web server could not send, and the server process fell over. In our lab, one anonymous request took the whole site down until it was restarted by hand. It was corrected in the next candidate and stayed corrected through release, but it is a neat illustration of why a fix has to be run, not just read: the code looked right.

How to update UP safely

1. **Take a backup first.** If you use mySites.guru, take a snapshot or run a full backup before you touch a live site.
2. **Update to 6.1.0 or 5.2.1.** On one site, use Joomla's System, Update, Extensions screen. Across several, use the mySites.guru mass updater to push it to every site at once.
3. **Confirm the version.** Check each site's extension list shows UP 6.1.0 (or 5.2.1 on older Joomla) or later.
4. **Look for files you did not put there.** The GitHub-install flaw wrote into the plugin's own folder, so check for action directories and PHP files under it that you do not recognise.
5. **Review who can write articles, and who is an administrator.** The author-level route ran for years of installs, and the admin-only actions are powerful by design.

Check your site for free →

How to find every site running UP

The first question after a disclosure like this is which of your sites run the thing. mySites.guru keeps a live inventory of the extensions, plugins and templates on each connected Joomla and WordPress site. Search for UP once and you see each site running it and the version each one is on.

We added the ranges from the CVE records, 5.0.0 to 5.2.0 on the older branch and every 6.x version below 6.1.0, to the mySites.guru vulnerability database, so every connected site still on a vulnerable UP is flagged on its dashboard without anyone searching. The same database lists the Joomla extension vulnerabilities we track, with the affected versions for each.

What else mySites.guru checks on a Joomla site

UP is one plugin. A typical Joomla site runs dozens of extensions, each with its own release notes and its own idea of what counts as a security release. No single check covers that. It takes several running together:

- Vulnerable extension detection compares installed extensions against the known-vulnerable ranges and flags a site as soon as one matches. That is how the UP rule reaches every connected site at once.
- Rogue administrator detection lists super users you did not create, which is where an abused admin-only action tends to leave its mark.
- File scanning checks every file against a pattern library of roughly 1,500 malware signatures and looks for files that can accept uploads, the kind a GitHub-install flaw could drop.

All of it runs unattended across every connected site, and all of it is part of the subscription. Doing the same by hand means logging into each Joomla admin, reading the extensions list and comparing it against changelogs, then doing it again the next time a vendor ships.

If a site has already been hit, fix.mySites.guru cleans it, audits it for backdoors and hands it back secure for a single fixed fee, usually the same day. Non-subscribers get a free month with it.

Start with a free audit on one site to see what it finds, or see the plans and connect every site you manage.

Fifteen candidates, and why we retested every one

Every build in this sequence arrived with a changelog saying the issues were fixed. Each time, part of it was true, part was not, and twice a build introduced something that had not been wrong before. One release normalised the action name in one place and closed four routes at once; the next dropped a line and reopened one. A permission check that read correctly crashed the site the first time an anonymous visitor reached it.

The maintainer, who had taken UP on after its author died and was not being paid for any of this, answered every report within hours and kept going through fifteen candidates until there was nothing left an author could reach. That is the outcome you want. It is also the reason the fix took four weeks: a vendor's changelog is a claim, not evidence, and the only way to know a security fix holds is to run the fixed build and try the attack again. We retested every candidate, and read the released 6.1.0 code against the last regressions before publishing this.

Disclosure and severity

We found these flaws reading UP 6.0.29's code and confirmed the anonymous file read and the author-level code execution live on our own test install. They were reported

privately on 23 August 2026, to the vendor address and the Joomla Security Strike Team, and nothing was published until a fixed release existed.

The Joomla CNA published four CVE records on 26 September 2026, one per weakness type, each crediting mySites.guru as the finder: CVE-2026-97163 for the remote code installation, CVE-2026-97160 for PHP command injection, CVE-2026-97161 for the path traversal and file access routes, and CVE-2026-97162 for the SQL injection routes. The weakness classes named in the records are path traversal (CWE-22) and improper access control (CWE-284), code injection (CWE-94) and SQL injection (CWE-89).

The scores below are the CNA's, worst first.

10.0
CVSS 4.0

CVE-2026-97163 Remote code installation from GitHub, no login needed

CRITICAL the Joomla CNA

An anonymous request set an on-demand code download in motion over a connection with certificate checks disabled, and whatever arrived was unpacked into a folder of PHP that UP runs.

`CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:N/VC:H/VI:H/VA:H/SC:H/SI:H/SA:H`

No login needed

▼ What does this mean?

CVSS 4.0 scores how severe a flaw is, from 0 (no impact) to 10 (critical). Each pair of letters in the string above is one metric; here is what this one's says.

How it is reached

`AV:N` Network: Reachable across the internet
`AC:L` Low: Nothing to work around, it just works
`AT:N` None: Works against any affected install
`PR:N` None: No account needed
`UI:N` None: Nobody has to be tricked into anything

What it does to the site

`VC:H` High: Everything the site holds can be read
`VI:H` High: Data and files can be altered at will

VA:H High: The site can be taken down

What it does beyond the site

SC:H High: Data on other systems can be read
SI:H High: Other systems can be altered
SA:H High: Other systems can be taken down

9.4
CVSS 4.0

CVE-2026-97160 PHP command injection through the php action

CRITICAL the Joomla CNA

The php action runs PHP from article text and is meant for administrators, which is why the CNA scores it as needing high privileges. Before 6.1.0 an article author could reach it by spelling the action name in a way the save check did not recognise, and the code ran for every visitor to the page.

`CVSS:4.0/AV:N/AC:L/AT:N/PR:H/UI:N/VC:H/VI:H/VA:H/SC:H/SI:H/SA:H`

Administrator action | Author-reachable before 6.1.0

▼ What does this mean?

CVSS 4.0 scores how severe a flaw is, from 0 (no impact) to 10 (critical). Each pair of letters in the string above is one metric; here is what this one's says.

How it is reached

AV:N Network: Reachable across the internet
AC:L Low: Nothing to work around, it just works
AT:N None: Works against any affected install
PR:H High: Needs an account with elevated rights
UI:N None: Nobody has to be tricked into anything

What it does to the site

VC:H High: Everything the site holds can be read
VI:H High: Data and files can be altered at will
VA:H High: The site can be taken down

What it does beyond the site

SC:H High: Data on other systems can be read
SI:H High: Other systems can be altered
SA:H High: Other systems can be taken down

9.2
CVSS 4.0

CVE-2026-97161 Arbitrary file read and file creation through path traversal

CRITICAL the Joomla CNA

One anonymous request to the AJAX endpoint returned any file the web server could read, including configuration.php and the database credentials it holds. Another wrote files through a path built from request input, outside the folder they belonged in.

`CVSS:4.0/AV:N/AC:L/AT:N/PR:N/UI:N/VC:H/VI:N/VA:N/SC:H/SI:N/SA:N`

No login needed | Exploitable over the internet

▼ What does this mean?

CVSS 4.0 scores how severe a flaw is, from 0 (no impact) to 10 (critical). Each pair of letters in the string above is one metric; here is what this one's says.

How it is reached

`AV:N` Network: Reachable across the internet
`AC:L` Low: Nothing to work around, it just works
`AT:N` None: Works against any affected install
`PR:N` None: No account needed
`UI:N` None: Nobody has to be tricked into anything

What it does to the site

`VC:H` High: Everything the site holds can be read
`VI:N` None: Nothing can be altered
`VA:N` None: The site stays up

What it does beyond the site

`SC:H` High: Data on other systems can be read
`SI:N` None: Other systems keep their integrity
`SA:N` None: Other systems stay up

8.3
CVSS 4.0

CVE-2026-97162 SQL injection vectors

HIGH the Joomla CNA

The sql action and actions that build queries from shortcode options passed attacker-controlled text into database queries. In one release candidate a subquery read password hashes from the users table.

`CVSS:4.0/AV:N/AC:L/AT:P/PR:N/UI:N/VC:N/VI:H/VA:H/SC:N/SI:N/SA:N`

Several routes

▼ What does this mean?

CVSS 4.0 scores how severe a flaw is, from 0 (no impact) to 10 (critical). Each pair of letters in the string above is one metric; here is what this one's says.

How it is reached

`AV:N` Network: Reachable across the internet
`AC:L` Low: Nothing to work around, it just works
`AT:P` Present: Needs a particular deployment or race to line up
`PR:N` None: No account needed
`UI:N` None: Nobody has to be tricked into anything

What it does to the site

`VC:N` None: Nothing can be read
`VI:H` High: Data and files can be altered at will
`VA:H` High: The site can be taken down

What it does beyond the site

`SC:N` None: Other systems keep their data
`SI:N` None: Other systems keep their integrity
`SA:N` None: Other systems stay up

Field	Detail
Component	UP, the “Universal Plugin” for Joomla (`plg_content_up`, plus the packaged action set)
Vendor	Originally Lomart, who died September 2025; maintained since by Pascal Le Conte (conseilgouz)
Type	Path traversal (CWE-22) and improper access control (CWE-284) for the GitHub install and file access routes; code injection (CWE-94) for the PHP action; SQL injection (CWE-89)
Affected versions	5.0 to 6.0.29 (per the maintainer)
Fixed in	UP 6.1.0 (Joomla 5.2 to 6.x) and UP 5.2.1 (Joomla 3.10 to 5.1), released 26 September 2026

Field	Detail
CVE	CVE-2026-97163 (remote code installation), CVE-2026-97160 (PHP command injection), CVE-2026-97161 (path traversal and file access), CVE-2026-97162 (SQL injection)
CVSS 4.0	Joomla CNA: 10.0 Critical (CVE-2026-97163), 9.4 Critical (CVE-2026-97160), 9.2 Critical (CVE-2026-97161), 8.3 High (CVE-2026-97162)
Finder	Phil Taylor, mySites.guru
Reported	23 August 2026, privately to the vendor and the Joomla Security Strike Team
Acknowledged	24 August 2026
Release reviewed	26 September 2026, code review of the released 6.1.0 build

Timeline

2025-09

UP's original author dies

Lomart, who wrote and maintained UP, died in September 2025. Pascal Le Conte took over maintenance. It was not announced anywhere we could find, and the contact page on the UP website returned a 404, which is why our report a year later went to an address no one was reading.

2026-08-23

mySites.guru audits UP 6.0.29 and reports four unauthenticated flaws

Arbitrary file read, file creation through path traversal, a file move and delete endpoint, and remote code installation. Reported the same night to the vendor address and the Joomla Security Strike Team, with CVE assignment requested through the Joomla CNA. The email also flagged that article authors could run PHP and SQL, without detail. All public detail withheld.

2026-08-24

The maintainer replies within ten hours

Pascal Le Conte acknowledged the report and committed to fixing it, and told us he now maintains the plugin. He also showed that the file move and delete endpoint never ran in 6.0.29 because of a PHP syntax error, which we confirmed, so three of the four were live and one was inert.

2026-08-25

First release candidate, and our first retest

A build arrived with a new setting reserving the risky actions to administrators. Retested the same day: one finding properly closed, two still exploitable, and the new setting could be walked around. We sent the author-level PHP and SQL findings in full.

2026-08-29

A build for the Joomla 3.10 to 5.1 branch joins in

Candidates for both the 6.1.0 and the 5.2.1 branch. Our retest found three of the original findings still reachable by the same route, and the changelog's claim to have fixed the SQL action did not hold up against the running code.

2026-09-01

The maintainer asks the Joomla CNA for CVE IDs

The SQL action was removed, and a different file-reading action, absent from the restricted list, gave an author the same read. Pascal asked the Joomla Security Strike Team to assign CVE identifiers and credit mySites.guru.

2026-09-07

The administrator-only list can be walked around four ways

The check reserving the risky actions compared the action name before the renderer rewrote it, and there were four rewrites, so there were four ways past it. Five further actions needed no trick at all, because they were never on the list.

2026-09-09

The structural fix arrives in RC11

A single normalisation function now works out the real action name for both the save check and the renderer, closing all four bypasses at once. An inverted file check and a denial-of-service crash are fixed too.

2026-09-11

RC12 regresses, and RC13 fixes it the same day

RC12 dropped the alternative-name lookup from that function and reopened one route, which we proved live and reported that morning.

RC13 restored the lookup and tightened the anonymous AJAX session check.

2026-09-14 — **The SQL action becomes administrator-only for good**

After we set out the remaining SQL injection paths, the SQL action was reserved to administrators, with queries touching password and token columns blocked.

2026-09-17 — **Our review of the last candidate finds nothing an author can reach**

Two small gaps remained in the save-time scan, both still blocked by the render-time check. The maintainer closed them in the final build.

2026-09-26 — **Four CVE IDs assigned**

The Joomla CNA assigned CVE-2026-97160 to CVE-2026-97163, grouping the vectors into one identifier per weakness type because each type shares a root cause and a fix.

2026-09-26 — **UP 6.1.0 and UP 5.2.1 released**

6.1.0 for Joomla 5.2 to 6.x, 5.2.1 for Joomla 3.10 to 5.1, held back until the CVE IDs were assigned. Every version from 5.0 to 6.0.29 is affected.

2026-09-26 — **Released build reviewed, affected sites flagged and this post published**

We read the released 6.1.0 code against the last regressions: the alternative-name lookup is back in the shared check, and the anonymous AJAX session check only accepts keys the plugin issued itself. The CVE records published the same afternoon, and every connected site on an affected UP version is flagged in mySites.guru.

Further Reading

- UP on GitHub - the maintainer's release page, where 6.1.0 is published.
- The Joomla extension security disclosure standard - what we ask vendors for, and why a fix that is not verified is not a fix.
- Why AJAX endpoints are a CMS security blind spot - the public-endpoint pattern behind three of these findings.
- A month of Joomla security disclosures - the wider picture across the ecosystem.
- CWE-22: Path Traversal, CWE-94: Code Injection and CWE-284: Improper Access Control - the weakness classes behind these findings.

Frequently Asked Questions

What was wrong with the UP plugin for Joomla?

UP, the Universal Plugin, turns {up ...} shortcodes inside article text into real work while the page renders: running PHP, querying the database, reading and writing files, and fetching code from remote servers. Around 160 actions ship with it. It also answers requests on Joomla's anonymous AJAX endpoint. In 6.0.29 that endpoint would read any file a visitor named, configuration.php included, create files outside the folder they belonged in, and trigger a download of action code from GitHub over a connection with its certificate checks switched off. Separately, the risky actions such as PHP and SQL were meant to be restricted to administrators, but a user who could write an article could reach them, and whatever they wrote ran for every visitor to the page.

Which versions are affected, and which are fixed?

The maintainer lists UP 5.0 to 6.0.29 as affected. The fix is UP 6.1.0 for Joomla 5.2 to 6.x, and UP 5.2.1 for Joomla 3.10 to 5.1. Anything older than 5.0 is outside the maintainer's stated range and has no fixed release of its own, so on those installs the safe move is to update to 5.2.1 or 6.1.0, or remove UP.

Who could exploit it?

Three of the flaws needed no login at all: one request to the site's AJAX endpoint was enough. The PHP and SQL actions needed permission to write an article, which on many real sites belongs to a client editing their own pages, a contributor, or a self-registered account on a site that allows front-end submission. Once a shortcode was saved, it ran for every anonymous visitor who opened the page, so the person who planted it did not need to stay logged in.

Can article authors still run PHP or SQL through UP 6.1.0?

No. 6.1.0 reserves the risky actions, PHP and SQL among them, to administrators, checks that when the article is saved and again when the page renders, and fixes the name-matching that earlier release candidates got wrong. The PHP and SQL actions still exist for administrators by design, so on a site running UP an administrator can read almost any database table from an article. That is a reason to treat who holds administrator access as carefully as you treat the server itself.

How do I know if my sites run UP?

Check each site's extension list for the content plugin named UP or Content - UP, and its version. mySites.guru keeps an extension inventory for every connected Joomla site and flags any site on a version in the affected range, so you see which sites need the update without logging into each one. If you manage sites for clients who write their own content, this is worth checking sooner rather than later.

Has this been exploited in the wild?

We have no evidence of that, and we are not aware of any public exploit. This was found in a private audit and reported privately, and nothing was published until a fixed release existed. This post describes the mechanisms and leaves out the working requests.

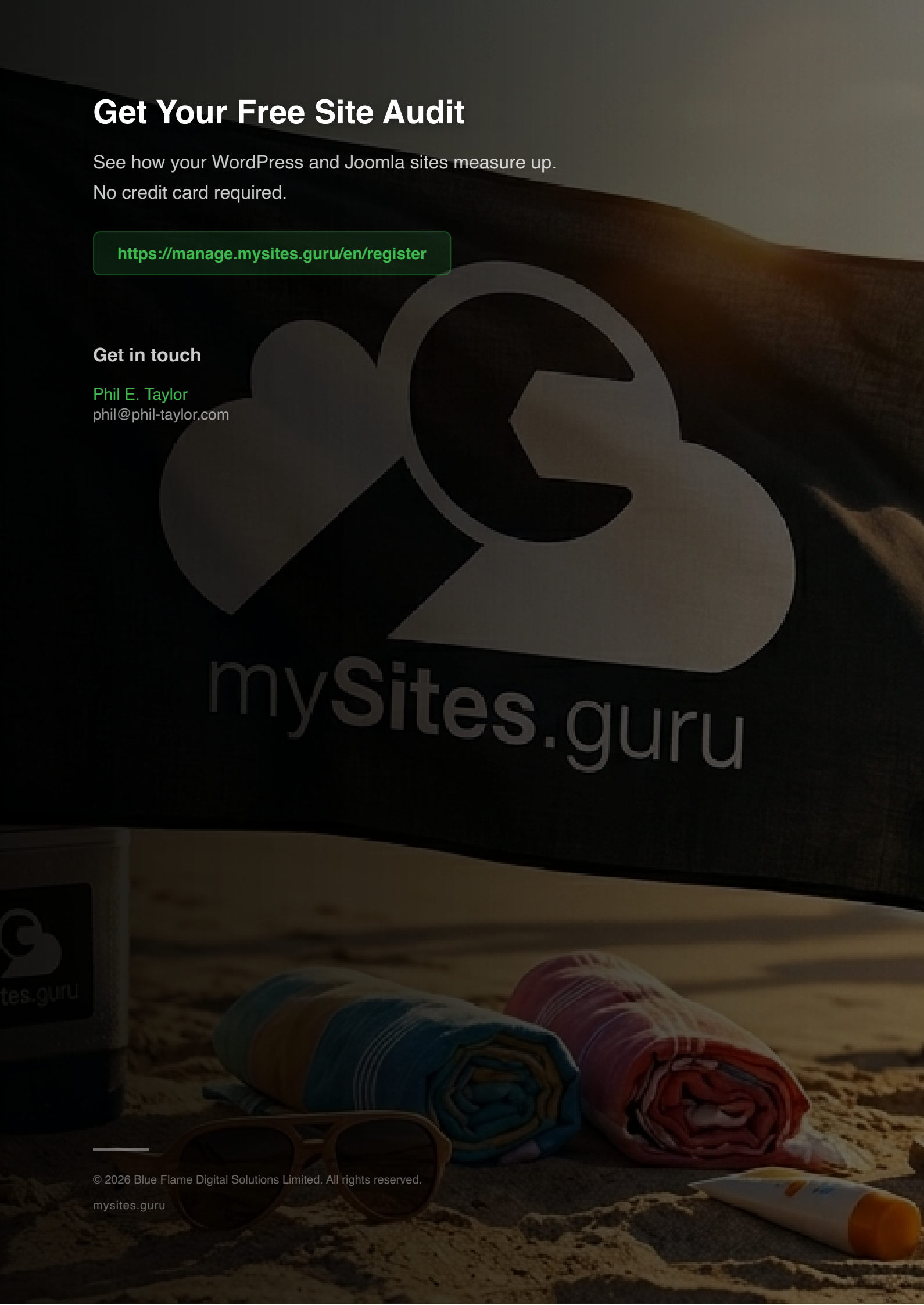
Written and edited by a human, with AI assistance. Our approach to AI

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