Unauthenticated SQL Injection in EDocman found by mySites.guru

mySites.guru found an unauthenticated SQL injection in EDocman for Joomla that let anyone read the whole database. Fixed in 3.9.0 - update now.

Phil E. Taylor | 15 July 2026

EDocman is one of the most widely installed document and download management extensions for Joomla. During routine security research on the extensions our customers rely on, **mySites.guru discovered an unauthenticated SQL injection vulnerability in EDocman and reported it privately to JoomDonation, who fixed it in EDocman 3.9.0.**

If you run EDocman on any Joomla site, update to 3.9.0 now. If you manage more than a handful of sites, read on for how to find every affected one at once. This is the same class of flaw we recently found and reported in AcyMailing and DPCalendar: a public front-end endpoint that trusts request input it should not.

mySites.guru discovered this security issue and reported it privately to the EDocman team. JoomDonation has now released the fix in EDocman 3.9.0. We are still withholding the exact request and any proof-of-concept while site owners update. This is how we handle every vulnerability we find: fix first, publish second.

TL;DR

- mySites.guru found an **unauthenticated SQL injection** in EDocman for Joomla and reported it responsibly to the vendor
- **EDocman 3.8 and earlier** are affected, and the vulnerable feature is long-standing. JoomDonation fixed it in **EDocman 3.9.0**
- A public front-end endpoint places a request parameter straight into a database query, so a single **anonymous request could read data from any table**: user accounts, password hashes, stored content, the Joomla secret, everything in the database

- We did not stop at theory. We **proved a full database read** on a test install, pulling back the site's Super User account and confirming the database connected as the MySQL superuser, so the read reached **every database on that server**
- **Update to EDocman 3.9.0 now.** If you cannot update immediately, make sure a SQL-filtering firewall is in front of the site as a stopgap
- We are still withholding the exact request and any proof-of-concept while site owners update
- mySites.guru flags every connected Joomla site still running a vulnerable version, so you do not have to check each site by hand

What is the vulnerability?

EDocman exposes a front-end endpoint that anonymous visitors can reach without logging in. That endpoint accepted a request parameter and placed it into a database query **without sanitising or quoting it**, the textbook shape of a SQL injection. Joomla's standard text filter, which the extension applied to that parameter, strips HTML but does nothing to neutralise SQL syntax: it removes tags and leaves quotation marks, parentheses and SQL keywords untouched. It is the same public-endpoint-meets-unchecked-input pattern behind a long run of recent Joomla extension vulnerabilities.

So an attacker could supply a crafted parameter that turned the query into one reading from any table in the Joomla database. No account needed, no CSRF token, nothing stolen first, and it works in a single request.

In practice that means an unauthenticated attacker could read:

- Joomla user accounts, including usernames and password hashes
- The Joomla configuration secret and password reset tokens
- Article and content records, extension configuration, session data and other stored values

We are deliberately not publishing the exact request, the affected endpoint or a working proof-of-concept. The point of this post is to get people onto 3.9.0, not to hand

attackers a recipe while sites are still updating.

This was a full database compromise, and we proved it

It is one thing to say a flaw "could" read the database. We confirmed it against a real EDocman install and read the data back. On a local test site running EDocman 3.8, a single anonymous request returned:

- the **database version** and the name of the live database
- the identity of the account the site connects to MySQL with, which was the **MySQL superuser** (`root`)
- the site's **Super User record** pulled straight out of the Joomla users table

That last point is the whole game. If an anonymous stranger can read the Super User row, they can read every row: every user account, every bcrypt password hash, every password reset token, the Joomla secret that signs sessions, and any personal data, order records or API keys stored anywhere in the database. Confidentiality is not partially affected here. It is gone.

There is an aggravating factor that makes this worse than one leaked site. On many Joomla installs, and on the one we tested, the database account the site connects with is **highly privileged**. Where that is the case, a read-only injection like this one is not limited to the one site's tables: it can read **every database on that MySQL server**, including the server's own internal user table that holds every account's hash. The blast radius of a single injection can be an entire shared database host, not just one Joomla site. That is why "it is only a read" is cold comfort. A read of everything is a compromise.

Error-based and blind: both routes work here

There are two broad flavours of SQL injection, and this endpoint is exposed to both.

The **fast** one is error-based. Where a site returns database errors to the browser, whether through Joomla's debug mode, PHP's `display_errors`, or a third-party error page or template that echoes the exception, an attacker forces the database to fail in a way that prints real data inside the error text. Each request comes back with dozens of characters of actual database content, so a Super User hash falls out in a handful of requests. We saw this fire on the test site even with Joomla debug switched off, because a third-party extension's error page leaked the database exception on its own.

The **quieter** one is blind. Where errors are fully suppressed, the response does not print anything useful, so the attacker asks the database true-or-false questions one at a time (is the first character of the admin hash greater than `m`?) and reads the answer from how the page behaves. It is slower, and it is not smaller. Automated tooling walks a database out one bit at a time in seconds of wall-clock, and the end state is identical: the full contents of every table an anonymous request can reach. Do not read "blind" as "limited". Blind is a throughput detail for the attacker, not a safety margin for you.

If you have not met blind SQL injection before, this short explainer from Indusface walks through the exact mechanics:

Watch video: https://www.youtube-nocookie.com/embed/8N7Z1aHFSIQ

None of this needs anything clever or new. sqlmap, the standard open-source tool, has automated both error-based and blind extraction and dumped entire databases point-and-click since 2006. A flaw a human would find tedious to exploit by hand is a single command that has been public for the better part of twenty years. The only reliable answer has always been the same one: update the affected code.

The changelog understates how serious this is

JoomDonation fixed this quickly, and released EDocman 3.9.0 the same day, which is genuinely to their credit. The problem is what the release notes tell a site owner. The security items in JoomDonation's own 3.9.0 release announcement read, in full (to see it yourself, open that product page and select the **About Latest Version** tab):

> *“Fixed a reported front-end SQL injection vulnerability.*
>
> *Improved database query handling and input validation.*
>
> *Strengthened security for public front-end tasks.*
>
> *Improved filtering for document, category, and date parameters.*
>
> *Added safer redirect validation to prevent open redirect issues.”*

That is the language of routine maintenance. “Improved input validation” and “strengthened security” are the words you use for a tidy-up, not for an emergency. A busy administrator skims that, files it under “housekeeping”, and leaves the update for next month.

Here is what those five lines actually describe. “A reported front-end SQL injection” is an **unauthenticated, no-login, single anonymous request that reads the entire database**, password hashes included, and on a privileged database account reads every other database on the server too. The changelog does not say unauthenticated. It does not say full database read. It does not say password hashes. Read on its own, it gives no reason to treat this as urgent, which is exactly the reason it is urgent: understatement is what keeps a critical hole open, because it persuades people not to patch.

The changelog is also quietly telling on itself. Those last two lines, “improved filtering for document, category, and date parameters” and “safer redirect validation”, confirm that the front-end SQL injection was **not the only issue in that code**. During the same audit we flagged additional raw request values reaching front-end queries, and an unvalidated redirect, and 3.9.0 hardened exactly those areas. One line in a changelog, several fixes underneath it. Treat 3.9.0 as the security release it is, not the maintenance release it reads like.

How serious is it?

Serious enough that we treated it as a priority disclosure. The two things that decide real-world impact are:

1. **Whether the site has a web application firewall that filters SQL.** Because the payload is literally SQL, a firewall that inspects request parameters can recognise and block many injection attempts before they reach EDocman. That is worth having, but it is a layer, not a guarantee: rules differ between products and configurations, injections can be encoded to slip past filters, and a firewall you have not tested against this specific flaw is an assumption, not a control. Sites running bare, with nothing inspecting requests, are the exposed ones.
2. **The site's own configuration**, including how much a database error reveals and how privileged the database account is. Where errors reach the browser, extraction is fast and loud; where they are suppressed, it is slow and quiet. Neither configuration makes the underlying flaw safe.

> This is the case for defence in depth. A SQL-filtering firewall in front of the site can buy you time against an injection like this, which is exactly why running one is worth it. It does not mean you can skip updating. Treat the firewall as the seatbelt and the patch as not crashing the car: you want both.

A firewall is mitigation, not a fix. The only reliable remedy is to update EDocman to 3.9.0.

Would a long, strong .htaccess have protected me against this?

Almost certainly not. A big hardened `.htaccess` is the security blanket a lot of Joomla owners reach for, and it guards a different kind of attack than this one. The exploit here is an ordinary, allowed request to `index.php`, the exact request your `.htaccess` exists to let through so the site works at all. The dangerous part is not the URL, it is the SQL hidden inside one query-string value, and a stock `.htaccess` never looks there.

To block this at the `.htaccess` layer you would need explicit rules that pattern-match SQL syntax in the query string and return a 403. Hardly anyone runs those, they are brittle, and they tend to block real visitors as often as attackers. Inspecting request input for SQL is really the job of a web application firewall with SQL injection filtering switched on, and even then only if that filtering is actually enabled. The short version applies here directly: update EDocman.

Which versions are affected?

EDocman **3.8**, the current release at the time of discovery, contains the affected code path. The vulnerable feature has been part of the extension for a long time, so earlier 3.x releases (and likely older builds) share it. Treat **any EDocman at or below 3.8** as affected. **JoomDonation fixed the issue in EDocman 3.9.0.**

> EDocman 3.9.0 fixes this. Update now, and do not wait to confirm exploitation: by the time you can confirm it, the data is already gone. If you cannot update immediately, make sure a SQL-filtering firewall is in front of any site running EDocman as a stopgap. This vulnerability is tracked as CVE-2026-57832.

How do you update EDocman safely?

JoomDonation released EDocman 3.9.0 with the fix. Update straight away. The steps are the same as any Joomla extension update:

1. **Take a backup first.** Before any extension update on a production Joomla site, back up the database and files. If you use mySites.guru, trigger a snapshot or a full backup so you have a clean starting point.
2. **Update through Joomla's Extensions manager, or in bulk from mySites.guru.** On a single site, open the Joomla administrator, go to System, then Update, then Extensions, and let Joomla pull EDocman 3.9.0. If it does not appear, use Find Updates, or download the latest package from your JoomDonation account and

install it over the top. If you manage more than one site, do not do this one panel at a time: use the mySites.guru mass update feature to upgrade every affected site from a single dashboard. We built the update runner to push Joomla extension updates to thousands of sites at once, and you can enable auto-updates for any Joomla extension so a fix like this reaches your sites without you lifting a finger.

3. **Confirm the version.** After updating, open Components, then EDocman, and check the version shown is **3.9.0 or later**.
4. **Clear caches.** Clear Joomla's cache and any CDN or page cache so stale front-end assets do not linger.

Updating closes the door. It does not undo any data an attacker may already have read, so if you handle sensitive data and were on a vulnerable version for a long time, treat exposed data and any stored credentials as potentially known.

How do I find every EDocman site I manage?

The first question after any extension security release is the awkward one: which of my sites actually run this? Up to about ten sites, you can log in to each Joomla admin and check the installed extensions list. Past that, you need a single view.

mySites.guru keeps a live inventory of every extension, template and framework on every Joomla and WordPress site in your account. You search for EDocman once and get back every connected site running it, the version each one is on, and whether an update is available. No logging into forty admin panels one at a time.

View every EDocman install across your sites

Open your Extension Inventory

Search for EDocman across every connected Joomla site and filter for anything on 3.8 or earlier to find the installs that still need the update. Not a subscriber? Sign up free and connect your sites.

Once you know which sites need it, the mass updater handles the rollout: tick the sites on an old version, push the update to all of them from one screen. When a fix ships, we add the affected range to our vulnerability database, so every connected site still on a vulnerable version of EDocman is flagged automatically. You should not have to learn about a Joomla security issue from the news and then go audit a portfolio of client sites by hand.

Why document and download extensions are a quiet attack surface

This is a pattern, not a one-off, and it is why the class of bug matters more than this single instance. Document managers, download managers and newsletter tools make attractive targets for the same reason they are useful: they expose front-end endpoints that anonymous visitors are meant to reach. Download links, category listings, unsubscribe links, search boxes, none of that can require a login.

Every one of those public endpoints is attack surface. When the code behind them trusts request input, or hands it to a database query without escaping, an anonymous visitor becomes an anonymous attacker. We have seen the same class of issue across newsletter tools like AcyMailing, calendar components like DPCalendar, form builders like Balbooa Forms, and page builders like PageBuilder CK: a public request that reaches a database query or a dangerous operation with too little checking in between.

The lesson for site owners is not “avoid document extensions”. They are fine, and the reputable ones are well maintained. The lesson is that **keeping them updated is security-critical, not just feature maintenance**. A document extension two versions behind is not a cosmetic problem, and as this changelog shows, a release that reads like housekeeping can be closing a hole in your whole database.

SQL injection is a recurring theme right now

This EDocman flaw is not an isolated find. It is one of several unauthenticated SQL injections we have uncovered in widely used Joomla extensions in a single research run,

and they all share one root cause: a public front-end endpoint that takes anonymous request input and hands it to a database query without turning it into a safe value first.

- AcyMailing (CVE-2026-56292, CVSS 8.7). The same class of unauthenticated SQL injection in one of the most popular newsletter extensions for Joomla, fixed in 10.11.1.
- DPCalendar (CVSS 8.7). The same shape again in a widely used calendar component, fixed in 10.11.2 and 8.19.4, with Digital Peak crediting us publicly for the find.
- The month of Joomla disclosures. EDocman is one entry in a larger run of vulnerabilities we found and reported across popular Joomla extensions in a single month, several of them this exact SQL injection shape.
- Why AJAX endpoints are a CMS security blind spot. The underlying pattern, and why front-end endpoints like this one keep producing the bug.

Stay ahead of the next one

This is one extension on one day. There will be another, because Joomla runs on thousands of third-party extensions and the ones that accept input from anonymous visitors keep producing bugs like this. The hard part is never the update itself. It is knowing a fix exists, knowing which of your sites are affected, and getting to them before an attacker does, across every extension on every site you look after.

That is the job mySites.guru does for you. It keeps a live inventory of every extension on every Joomla and WordPress site in your account, flags the ones with a known vulnerability, and lets you push the update to all of them from one screen. When something like this EDocman flaw is disclosed, you see exactly which sites are exposed in seconds, without opening a single admin panel to check.

Subscribe to security alerts

Want the alerts and the tooling to act on them? Start with a free audit on one site and see your full extension inventory, or sign up for mySites.guru to get vulnerability alerts and one-click updates across every site you manage.

What to do right now

- Update every Joomla site running EDocman to **3.9.0** or later, one at a time or in bulk from one dashboard
- If you cannot update immediately, make sure a SQL-filtering web application firewall is in front of any site running EDocman, and remember it is mitigation, not a fix
- If you manage multiple sites, let mySites.guru show you which ones are still exposed rather than checking by hand
- Take a backup before updating, and clear caches afterwards
- If you were on a vulnerable version for a long time and handle sensitive data, treat any exposed data as potentially read, rotate exposed credentials and the Joomla secret, and if you suspect a breach, find any hacked files and backdoors and follow our guide to fixing a hacked Joomla or WordPress site

Disclosure and Severity

This flaw is CWE-89, SQL injection, reached by an anonymous visitor over the network in a single request, with no privileges and no user interaction. We score it **CVSS 4.0 8.7 (High)**. It is a read-only injection, so it stops short of the maximum 10.0 you see on file-upload flaws that end in remote code execution: an attacker reads the database but cannot write to it or run code through this flaw. Reading the database is bad enough, because what leaks is every user record, every password hash and the Joomla secret. Where the site's database account is privileged, as it was on the install we tested, the

practical impact is higher still, because the read reaches every database on that MySQL server, not just the one site.

8.7
CVSS 4.0

HIGH **Our own assessment, pending an official score**

Unauthenticated, exploitable over the internet in a single request with no user interaction, ending in full read access to the site database. It falls short of 10.0 only because it is a read, not a write: high confidentiality impact, but no integrity or availability impact.

- No login needed
- Exploitable over the internet
- No user interaction
- Full database read
- Password hashes exposed

Field	Detail
CVE	CVE-2026-57832 (assigned via the Joomla CNA)
Component	EDocman for Joomla (`com_edocman`)
Vendor	JoomDonation / Ossolution (joomdonation.com)
Type	Unauthenticated SQL injection (CWE-89)
CVSS 4.0	8.7 (High), `AV:N/AC:L/AT:N/PR:N/UI:N/VC:H/VI:N/VA:N/SC:N/SI:N/SA:N` (our own assessment)
Impact	Anonymous read access to any database table, including user accounts and password hashes
Finder	Phil Taylor, mySites.guru
Affected versions	3.8 and earlier
Fixed in	3.9.0 (released 14 July 2026)

We follow a fix-first, publish-second process: private disclosure to the vendor, a window for a patch, then a public write-up without a proof-of-concept.

Date	Event
12 July 2026	During routine security research on the extensions our customers rely on, mySites.guru identifies the unauthenticated SQL injection in EDocman 3.8, confirms it against the code, and proves it on a local test install that reads database contents back through a single anonymous request. The issue is disclosed privately to JoomDonation, with all public detail withheld.
14 July 2026	JoomDonation releases **EDocman 3.9.0**, fixing the reported front-end SQL injection along with tighter input validation on document, category and date parameters and safer redirect validation. mySites.guru verifies the fix, adds the affected range to its Joomla vulnerability database so every connected site below the fix is flagged, requests a CVE through the Joomla CNA, and publishes this write-up. No proof-of-concept is released.
15 July 2026	The Joomla CNA assigns CVE-2026-57832 to the vulnerability, crediting Phil Taylor of mySites.guru as the finder.

Further Reading

- This EDocman flaw was one of a month of Joomla extension vulnerabilities we found and disclosed - the full roundup of the run.
- Our DPCalendar SQL injection disclosure and our AcyMailing SQL injection disclosure - the same class of flaw in other Joomla extensions, found in the same period.
- Why AJAX endpoints are a CMS security blind spot - the recurring public-endpoint pattern behind flaws like this one.
- CWE-89: SQL Injection and OWASP: SQL Injection - the canonical references for this weakness class.
- EDocman product page and 3.9.0 release announcement - where 3.9.0 is available to download; open the **About Latest Version** tab to read JoomDonation's own release notes.

Frequently Asked Questions

What is the EDocman SQL injection vulnerability?

It is an unauthenticated SQL injection in EDocman, the document and download manager for Joomla by JoomDonation. A public front-end endpoint took a request parameter and placed it into a database query without sanitising it, letting an anonymous visitor read data from any table in the site's database. mySites.guru discovered the issue and reported it privately to the vendor. It is fixed in EDocman 3.9.0.

Do I need to log in to Joomla for an attacker to exploit this?

No. The affected endpoint is reachable by anonymous visitors in a single request, so an attacker needs no account, no CSRF token and no stolen credentials. That is what makes it serious. The main real-world limits are whether the site sits behind a web application firewall that filters SQL, and the site's own configuration.

Which EDocman versions are affected?

EDocman 3.8, the current release at the time of discovery, contains the affected code path, and the vulnerable feature is long-standing, so earlier 3.x releases should be treated as affected too. If you run any version at or below 3.8, assume you are affected. JoomDonation fixed it in EDocman 3.9.0, so update to 3.9.0 or later straight away.

How do I know if my site is affected?

If you run EDocman on a Joomla site at or below version 3.8, assume you are affected and update to 3.9.0. mySites.guru tracks the installed extensions and versions on every connected site, so every connected site running a vulnerable version of EDocman is flagged automatically. You get told which sites need attention rather than checking each one by hand.

Was this exploited in the wild before the fix?

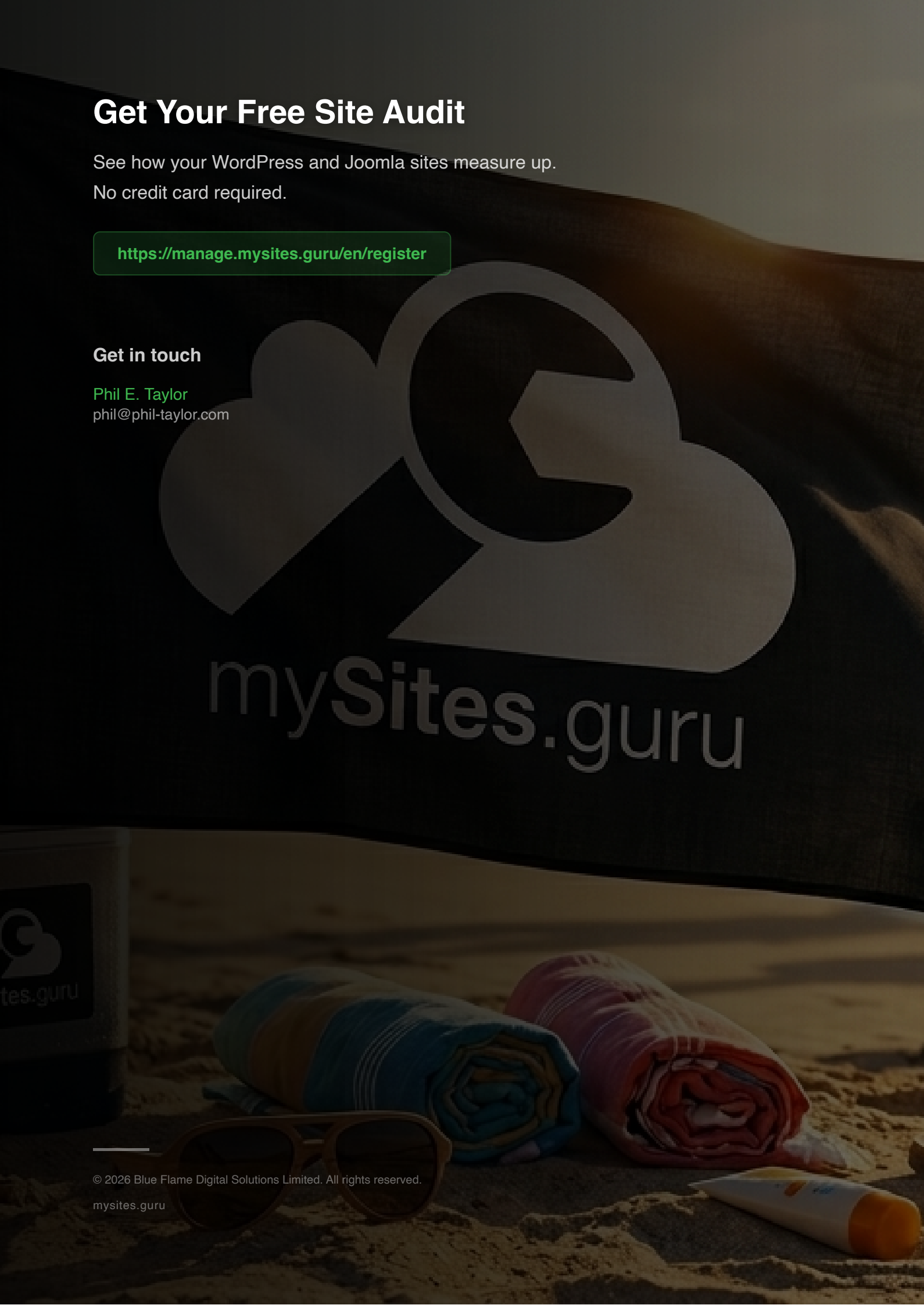
We have no evidence of exploitation. We disclosed the issue privately to JoomDonation, who have now released the fix in EDocman 3.9.0, and we are still withholding the exact request and any proof-of-concept while site owners update. Because the payload is SQL, a web application firewall that filters SQL may block it, but a firewall is mitigation, not a cure.

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