0-DAY NO CVE HIGH PUBLIC

phpMyAdmin setup/validate.php live after install

phpMyAdmin 5.2.3 setup/index.php dies when config.inc.php exists. setup/validate.php does not. Unauth POST with a login-page token opens a MySQL-protocol TCP connect to attacker host:port. Labbed.

NAME phpMyAdmin setup/validate.php live after install	TYPE 0-day analysis
CVE n/a	CVE RISK high
DISCLOSURE STATUS public	VENDOR phpMyAdmin
AFFECTED phpMyAdmin 5.2.3 (962857e) when setup/ is still in the docroot. Official phpmyadmin:5.2.3 image deletes setup/. PMASA-2016-44 did not apply the config-exists fatal to validate.php.	PUBLISHED 12 Oct 2026
UPDATED 12 Oct 2026	TAGS 0-day, phpmyadmin, ssrf, setup, cwe-918, unauthenticated

TL;DR

Lab files and proof-of-concept **here**.

01 Setup is disabled. Validate is not.

I am @abraxas_null. The proof of concept is on GitHub: **abraxas/phpmyadmin-setup-validate** (loopback client). The lab stack is `lab/`: Dockerfile, docker-compose.yml, run.sh, poc.py. Authorized lab only. It talks to loopback.

This is **phpMyAdmin 5.2.3** (`962857e`). No CVE yet. CWE-918. 7.2 High (`CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:L/I:L/A:N`). Remote SSRF over the MySQL protocol. Unauthenticated if `setup/` is still web-reachable. No MySQL login. CSRF token from the public login page.

`setup/index.php` dies when `config.inc.php` is present. `setup/validate.php` does not. It still loads under `$isMinimumCommon`, takes a token from `GET /`, and calls `Validator::testDBConnection` → `mysqli_connect` to the host, port, user, and password in the JSON `values` field. The phpMyAdmin host opens a TCP MySQL handshake to that address.

The official `phpmyadmin:5.2.3` image deletes `setup/`. This leftover only applies when that directory is still in the web root, which is the common leftover after an install. The lab restores `setup/` from the 5.2.3 tarball so the pin matches `RELEASE_5_2_3`.

PMASA-2016-44 / CVE-2016-6621 covered some setup SSRF. The "disable setup when config exists" check was never applied to `validate.php` / `config.php`.

02 Why setup after install

Same wave as WebAuthn. Auth/CSRF/setup was the hunt. Setup is supposed to be a first-run wizard. After `config.inc.php` exists, the wizard is supposed to die. One

script got the fatal. The validator did not. That is a leftover of a control, not a new SSRF primitive.

I only wanted Critical or High. HTTP metadata SSRF would be a different class. This handshake is MySQL. Internal 3306, credential stuffing against an internal server, a port oracle from timing and mysqli errors. Scope C because the PMA host is talking to an address the attacker chose. C:L I:L because the protocol is MySQL, not GET `/latest/meta-data`.

03 Pinning RELEASE_5_2_3

Latest 5.2 stable. Tag `RELEASE_5_2_3`. Commit `962857e`. Official image plus `setup/` restored from the 5.2.3 tarball. `config.inc.php` present. Setup UI disabled. Validate still live. TCP oracle on `oracle:13306`.

04 The skip list is half the hunt

- PMASA-2025-1 / 2 / 3 already patched in 5.2.2.
- PMASA-2016-44 / CVE-2016-6621 some setup SSRF. The config-exists fatal was the follow-up. It landed on `index.php`. It did not land on `validate.php`. Do not re-file 2016. File the script that missed the fatal.
- Official `phpmyadmin:5.2.3` image deletes `setup/`. That is a deployment that does not have this leftover. FAIL if `setup/` is not deployed. Not a reason to skip tarball installs that left the directory.
- WebAuthn, pmadb SQLi, CSRF `/sql` are different leftovers on this pin.

I am not filing "setup wizard can connect to MySQL." That is the wizard. I am filing the wizard's validator after the wizard is disabled.

05 Surfaces that returned nothing

File/LFI, deser, transformations, changelog leftovers at this bar came back empty or Medium. Hide-db `validateRegex` interpolates a user regex into `preg_match`. That is ReDoS. I did not run it. Inferred, not this SUCCESS. ACME-style HTTP fetch to metadata is not this handshake.

`setup/lib/common.inc.php` sets `$isMinimumCommon` and does not look for `CONFIG_FILE`. Token check still runs. POST without a token does not hit the oracle. That is the negative.

06 The fatal that only one script has

`setup/index.php`:

PHP Copy

```php
if (@file_exists(CONFIG_FILE) && ! $cfg['DBG']['demo']) {
    die('Configuration already exists, setup is disabled!');
}
```

`setup/validate.php` has no that check. `Validator::testDBConnection` / `mysqli_connect` (host, user, password, port, socket from JSON `values`). Secondary: `validateRegex` builds `preg_match('/' . $values[$path] . '/', ...)`.

Unauthenticated client GETs `/`, picks up a session cookie and token, POSTs `setup/validate.php` `id=Server` with `Servers/1/host` and `Servers/1/port`. The PMA host connects.

07 The lab

Loopback `http://127.0.0.1:19230`. Compose project `phpmyadmin-setup-validate`. Dockerfile restores `setup/` from the 5.2.3 tarball and runs a TCP oracle on port 13306.

`GET /setup/index.php` still said setup is disabled. POST without a token did not hit the oracle. POST with the login-page token connected to `oracle:13306` and wrote `PMA-SETUP-VALIDATE-WITNESS`. Body was `mysqli_connect(): (HY000/2006): MySQL server has gone away` after the oracle accepted then closed. Peer was the PMA container.

PLAIN TEXT Copy

```
cd lab && ./run.sh
```

Last line:

PLAIN TEXT Copy

```
SUCCESS phpmyadmin-setup-validate how=live tcp=yes setup-index-disabled=yes PMA-S
```

08 Wrong turns, in the order I made them

Treating the official image as this leftover. It deletes `setup/`. Restore the directory from the tarball or you are labbing Docker Hub, not `RELEASE_5_2_3`.

POSTing without the login-page token. Zero oracle hits. The token check still runs under `$isMinimumCommon`. Get the token from `GET /`. You do not need a MySQL login.

Treating this as HTTP metadata SSRF. The handshake is MySQL. Something on that port has to speak MySQL or you get a protocol error. Internal 3306 is the interesting case. `169.254.169.254` is not, unless it answers MySQL.

Running the ReDoS path and calling it this leftover. `hide_db` regex is inferred. The SUCCESS is TCP.

A reverse shell. Theatre. The oracle is a TCP accept log on loopback.

09 What an attacker can do

Anyone who can reach a leftover `setup/` tree can make the phpMyAdmin host open a MySQL-protocol TCP connection to a host and port they choose.

PRIVILEGE
Open TCP from the PMA host to attacker host:port (integrity of outbound connect; recon of th
Confirm setup UI is off while validate stays on
Fail closed without the login-page CSRF token
Port / credential oracle from mysqli errors and timing

PRIVILEGE
Talk MySQL to an internal 3306 and try posted user/password
ReDoS via `validateRegex` interpolating user regex

Closed on this lab: HTTP metadata SSRF, unauthenticated PMA login, OS shell, and the setup wizard itself (`index.php` stays disabled). FAIL if `setup/` is not deployed.

I am not going to print a host:port you can paste at someone else's PMA. If you own the instance, run the proof of concept against loopback.

10 The fix

Put the `file_exists(CONFIG_FILE)` fatal in `setup/lib/common.inc.php` so **every** setup script dies. Restrict `testDBConnection` hosts. Do not build `preg_match` delimiters from user strings. Delete `setup/` from the docroot after install, which the official image already does.

Same pin leftovers: phpmyadmin-webauthn, phpmyadmin-pmadb-central-sqli, phpmyadmin-designer-json-sqli, phpmyadmin-csrf-sql.

11 References

- Proof of concept: abraxas/phpmyadmin-setup-validate · phpmyadmin-setup-validate-Abraxas-Labs.py
- Lab: `lab/` · Dockerfile · docker-compose.yml · run.sh · poc.py
- @abraxas_null · github.com/abraxas · abraxaslabs.tech · abraxas.null@proton.me
- CWE-918 · 7.2 High
- Tree: phpmyadmin/phpmyadmin RELEASE_5_2_3 (`962857e`) · `setup/index.php` · `setup/validate.php` · `Config/Validator.php` `testDBConnection`

Abraxas Labs

Independent security research by abraxas.

Analysis of 0-days, n-days, malware, and the leftovers that still matter.

GitHub **@abraxas_null** **abraxas.null@proton.me** **abraxaslabs.tech**

Home Research Open Source Reading Room About GitHub @abraxas_null Console

Independent research. Private analysis by arrangement. No ads. abraxas.null@proton.me · For defensive awareness and authorized testing only.

© 2026 Abraxas Labs