MAMP Audit: Database Review + Web Stack Vulnerability Remediation (No DB Changes)

Last Updated: 2026-02-17

Scope

This document records a **read-only terminal audit** of the local MAMP environment and defines how to remediate the reported PHP/Apache vulnerabilities **without modifying database schemas, tables, procedures, triggers, or data**.

Audit Method (Read-Only)

The following actions were performed from terminal only:

- Located MAMP binaries under `C:\MAMP`.
- Queried MySQL metadata using `SHOW DATABASES` and `information_schema`.
- Checked running Apache process command line.
- Read active Apache config from `C:\MAMP\conf\apache\httpd.conf`.
- Collected HTTP response headers via `curl -I`.

No DDL/DML statements were executed.

Live Environment Findings

Web Stack

- Apache binary version: `Apache/2.4.33 (Win64)`
- PHP exposed in headers: `PHP/8.3.1`
- OpenSSL in server banner: `OpenSSL/1.0.2u`
- Server header currently exposes detailed versions because:
  - `ServerTokens Full`

- `X-Powered-By: PHP/8.3.1` is present in responses

Scanner-visible evidence (from local headers)

```
curl -I http://127.0.0.1
```

- `Server: Apache/2.4.33 (Win64) OpenSSL/1.0.2u mod_fcgid/2.3.9 PHP/8.3.1`
- `X-Powered-By: PHP/8.3.1`

This directly explains why unauthenticated vulnerability plugins continue to flag the host.

MySQL Server

- MySQL version: `5.7.24`
- Port: `3306`

Database-by-Database Audit Summary

Application/user databases discovered:

1. `easy_inventory`
2. `mtm_receiving_application`
3. `mtm_wip_application_winforms`
4. `mtm_wip_application_winforms_test`

System schemas also present: `information_schema`, `mysql`, `performance_schema`, `sys`.

1) easy_inventory

- Base tables: `1`
- Views: `0`
- Routines: `0`
- Triggers: `0`
- Approx. size: `0.02 MB`

2) mtm_receiving_application

- Base tables: `27`
- Views: `4`
- Routines: `141`

- Triggers: `2`
- Approx. size: `1.42 MB`

3) mtm_wip_application_winforms

- Base tables: `29`
- Views: `0`
- Routines: `170`
- Triggers: `0`
- Approx. size: `8.41 MB`

4) mtm_wip_application_winforms_test

- Base tables: `29`
- Views: `0`
- Routines: `170`
- Triggers: `0`
- Approx. size: `5.44 MB`

Key Conclusion About Databases

The reported vulnerabilities (#157, #160, #161) are in the **web serving layer (Apache/PHP exposure and version range)**, not in MySQL objects.

Therefore, fix strategy is:

- **Do not alter any databases** (`CREATE/ALTER/DROP/INSERT/UPDATE/DELETE` not required).
- Patch and harden **Apache/PHP configuration and binaries**.

Remediation Plan (No DB Changes)

Step 1 — Backup only config/runtime artifacts

- Backup `C:\MAMP\conf\apache\httpd.conf`
- Backup active PHP INI referenced by Apache (`C:\MAMP\conf\php8.3.1\php.ini`)
- Backup virtual host and SSL files if used

Step 2 — Upgrade vulnerable runtime components

- Upgrade Apache from `2.4.33` to current supported 2.4.x (latest stable preferred).
- Upgrade PHP from `8.3.1` to at least `8.3.19` (or newer patched branch).

Note: ticket guidance references 8.3.19/8.2.28/8.1.32/8.4.5 patch lines for PHP.

Step 3 — Header hardening (still no DB impact)

In active Apache config:

- Change `ServerTokens Full` → `ServerTokens Prod`
- Keep/ensure `ServerSignature Off`

In active PHP INI:

- Set `expose_php = Off`

Step 4 — Minimize risky modules if unused

Given tickets mention module-related CVEs in old Apache:

- Review whether `mod_isapi`, `mod_lua`, proxy modules are truly required.
- Disable unused modules in Apache config.

Step 5 — Restart and validate

- Restart Apache/MAMP services.
- Re-check headers:
  - `curl -I http://127.0.0.1`
- Confirm new versions:
  - `httpd -v`
  - `php -v` (active Apache PHP runtime)
- Re-run vulnerability scan.

External Evidence (Internet Research)

The following publicly available sources support this remediation approach and explain why these findings are common in bundled stacks like MAMP/XAMPP:

1. Apache official vulnerability list confirms affected old Apache ranges and fixed versions:
   - https://httpd.apache.org/security/vulnerabilities_24.html
2. PHP official downloads and changelogs show patched release lines and security fixes:
   - https://www.php.net/downloads
   - https://www.php.net/ChangeLog-8.php
3. Tenable/Nessus plugin pages indicate many detections are based on **self-reported version/banner**:
   - https://www.tenable.com/plugins/nessus/161454
4. Community discussion patterns (search results) repeatedly point to outdated bundled stacks being flagged and resolved via upgrade/hardening:
   - https://duckduckgo.com/?q=MAMP+Apache+2.4.53+vulnerability
   - https://www.reddit.com/search/?q=MAMP apache vulnerability

Operational Notes

- This remediation can be executed with **zero database object changes**.
- Application smoke tests should focus on web app behavior and DB connectivity only.
- If scan findings persist after upgrade, check for:
  - stale reverse proxy or load balancer headers
  - multiple Apache instances
  - old binaries still in service path

Ticket Response Snippet (No-DB-Change Statement)

We performed a read-only audit of all MAMP application databases and confirmed the reported finc

Databases reviewed: easy_inventory, mtm_receiving_application, mtm_wip_application_winforms, mtr
No database changes are required for remediation.

Remediation is limited to web stack patching/hardening:
- Upgrade Apache to current supported 2.4.x
- Upgrade PHP to patched release line (8.3.19+ or newer)
- Harden headers (ServerTokens Prod, ServerSignature Off, expose_php Off)
- Re-run vulnerability scan and attach evidence