OpenEMR Weekly Meeting Summary

August 4, 2026

Attendees included: Stephen Nielson (chair), Stephen Waite, Eric Stern (OCE), Sherwin Gaddis, David Eschelbacher, Greg Sirotek (OCE). Brady Miller and Michael Smith absent.

Vulnerability Disclosure and Severity Assessment

- **A researcher published a set of vulnerabilities publicly rather than through the GitHub advisory process**, after growing frustrated with how long triage was taking and with the outcome of one of the advisories.
- CVEs can be obtained through issuing agencies other than GitHub, so the project does not control who receives a CVE or what severity is assigned.
- Two of the original reports had already been triaged by the project — one had been previously reported with the fix still outstanding, the other was judged not addressable within OpenEMR itself. Additional reports were filed directly with an outside issuer.
- **The project's review found that several of the published items depend on an administrator deliberately enabling an off-by-default feature or granting a third-party application access.** These are configuration decisions rather than defects, and the project's assessment of their severity differs from the rating assigned.
- Anonymous application registration, cited in one report, is a requirement of the regulatory framework OpenEMR operates under rather than an oversight.
- **Severity inflation is a growing industry-wide problem.** The curl project now maintains its own severity assessments for the same reason — outside issuers over-assign severity, and reviewers are too overwhelmed to push back.
- **Practical consequence:** a high-severity CVE creates contractual patching obligations for downstream users regardless of the project's own assessment, so fixes have to be produced either way.

How the Project Plans to Respond

- Responding to the issuer directly is low cost but unlikely to change an assigned rating.
- **A short blog article was proposed** explaining the project's approach to severity assessment, with practical guidance for affected configurations — something to reference the next time this comes up.
- Add clearer warnings to features that ship disabled, so operators understand the risk before enabling them.
- Expect more of this as the backlog grows. Faster triage and disclosure would reduce the pressure, and work is underway to automate parts of advisory reconciliation.

PR 13374 — OAuth Client Identity in the API Audit Log

Pull request: https://github.com/openemr/openemr/pull/13374

- **Adds a distinct client ID field to the audit log** rather than continuing to record it in the user ID column.
- Open question to verify: the client ID may already be populated into the user ID field in the API response logger listener, possibly only for system clients using the client credentials flow rather than the authorization grant, where a real user account exists.
- **Agreement that keeping the two values separate is the right approach**, with a check for where the existing value is already relied upon.
- Context: the audit log tables were substantially refactored a few months ago — a control flow change rather than a change to the data recorded.
- **Additional request: expose client ID as a filter in the audit log interface**, not just record it, so operators can review activity by client.
- A log parser to make audit data easier to work with is partially built and would help here; current logs are difficult to query quickly.

AI Integration and the Community Forum Thread

Forum thread: https://community.open-emr.org/t/feedback-requested-generic-oidc-login-and-ai-integration-for-openemr/26915

- **Two community members are working on AI integration from different angles** — one on the client side, the other on the supporting infrastructure. The original poster's generic OIDC login approach aligns more closely with what the project could integrate.
- **The thread has been open a couple of weeks without a response.** The work is considered important and getting a reply posted was treated as the immediate action; the delay reflects summer availability and current workload rather than lack of interest.

The Identity Provider Question

- The upstream League OAuth2 server expects the identity provider and the authorization server to be the same, which makes third-party identity providers difficult. There is OpenID Connect work upstream, but OpenEMR is largely on its own to support this.
- The related upstream issue has been open for roughly four years, so the project should not plan around an upstream fix.
- A community PR was noted attempting to extend this so an external identity provider (Google, Keycloak, or similar) could be used with OpenEMR acting as the authorization server. Status to be checked.
- Assessment: the four existing grants — authorization code, client credentials, device authorization, and refresh — are agnostic to the identity provider, so a new custom grant is probably not required. What needs to change is how the identity provider is handled in the workflow. OpenEMR already carries several custom grants because upstream does not support them.
- The pattern of interest: authorization completed through an external provider, then handed to an AI agent via token exchange — the distinction between delegation and impersonation.

Scoping Access for AI Agents

- **OpenEMR's ACLs are broad, and there is no filtering of specific data sets within a broad ACL today.** Narrowing what an integration can reach would require either more granular ACLs throughout the codebase or changes to the permission model.
- Operators should be deliberate about which grant type they issue to an integration and should scope access to the minimum the integration requires.
- Why agents raise the stakes: an automated client can operate far faster and more persistently than a human user, so the same access carries more risk.
- **Direction of travel:** centralize database access into repositories so requests flow through a single point where data can be checked. A data filtering mechanism already exists on the API that filters response sets against resources and ACLs, and an initial policy enforcement layer has been built.
- **Doctrine is the enabler** — consolidating on it makes it far more practical to apply controls at the database layer. This connects directly to the front controller work: a single entry point is what makes centralized checks possible.
- Longer-term needs include more granular ACLs and tracking of patient and provider consent. Policy evaluation at the PHP layer is slow, which is a known constraint on this approach.
- Realistic expectation set: this will not be solved in the next few months, but progress is steady.

Tagging Clients and Users

- **Proposal: support tags on user accounts and clients, including an AI tag, and allow the audit log to be filtered by tag.**
- Implementation is straightforward — identify the users carrying a tag, then retrieve matching audit entries.
- The value is traceability. There is currently no way to distinguish an automated client from a conventional one beyond logging which client made the request.
- Tagging user accounts was considered useful generally, beyond the AI case — for example, filtering audit activity for any defined group of users.

Development Practices Around AI Tooling

Referenced during the discussion: https://xkcd.com/2347/ — noted as being based on the maintainer of curl, a single individual sustaining a dependency much of the internet relies on.

- **The practical question: how to use AI development tooling without risking exposure of PHI or client-confidential code.** Running highly restricted produces enough confirmation prompts that people stop reading them; running unrestricted risks premature pushes and leakage from private repositories.

Practices Shared

- **Keep PHI off the development machine entirely** — the simplest and most effective control available.
- **Tune permissions deliberately** — auto-allow routine, non-destructive commands so that the prompts that do appear are meaningful, and require confirmation on anything destructive.
- **Use enforcement hooks rather than written instructions.** Guidance placed in a project README tends to be ignored; a hook is enforced at call time and can return a rejection message, which causes the agent to adjust its approach. One example: auto-rejecting long chained shell commands, which reliably produces simpler and more reviewable actions.
- Require confirmation on commit operations so nothing enters version control unreviewed.
- Outbound traffic filtering with path allowlisting was raised as an option for those with stricter requirements.

Threat Model Notes

- **Supply chain prompt injection is the more realistic concern** — untrusted content reaching an agent session and influencing its actions. This is documented behavior across the industry, not a theoretical risk. A useful comparison is package manager post-install hooks, where checking out a repository and running an install can compromise a machine if configured carelessly.
- Comprehensive prevention of outbound data leakage requires enterprise network controls that most contributors do not have available.
- Confidential source code is a real concern, but database content is the higher one — and that is a threat vector with decades of established practice behind it. The same controls largely apply.
- **The broader concern raised:** the pressure to trade security for clinical workflow convenience is significant, and an integration that is convenient enough will get adopted whether or not it is scoped safely. This is why access scoping and audit traceability matter now rather than later.
- Counterweight noted: the current rate of vulnerability discovery is forcing fundamental fixes that had previously been deferred.

Removing backup.php

- **Recommendation on the table: remove backup.php from core due to recurring security issues.**
- **What it does:** provides a downloadable copy of the database and demographic data, intended for operators without direct access to their own server.
- **Why it keeps producing issues:** the feature's purpose is bulk data export, so any weakness in the access control around it results in exactly the outcome the feature was built to perform. Multiple rounds of fixes have been applied and further issues have continued to surface.
- **Recommended replacement:** standard database backups plus a separately stored copy of documents — rsync and mysqldump are the appropriate tools. Environments where those are not available are not appropriate environments for production OpenEMR.
- Position stated plainly: operators handling PHI need a working backup strategy independent of any single application feature.

Removal vs. Archived Module

- **Concern raised:** removing features from a product that users already consider feature-limited is risky. If something must leave core, moving it to a module is preferable to deleting it.
- **Counter-position:** for high-risk, low-usage code, removal is cleaner, and installations far enough out of date to still depend on it are generally not upgrading anyway.
- **Middle path discussed:** extract the feature into a module marked archived and explicitly unsupported — available to fork and maintain, with no support commitment from the project.
- **A forum post was published during the meeting** covering recommended backup practices for production data and inviting anyone interested in maintaining the feature as a module to step forward.
- **The governing principle:** the deciding question is not whether a feature has value but who is available to maintain it. If someone will take on the security and maintenance work, the feature can live on as a module.

The Broader Core-to-Module Direction

- Consistent with prior discussions: move functionality out of core and into modules. A module's risk surface is limited to the installations that choose to run it.
- Secondary benefit: several large subsystems account for a disproportionate share of static analysis time, so moving them out of core speeds the toolchain for everyone.
- Acknowledged cost: some of this functionality is genuinely valued — OpenEMR's ophthalmology support, for example, is well regarded within that specialty — so removals impose a real burden on specific communities.
- Upside: specialty-focused modules let a practice run only the functionality it actually needs.
- Caveat: the assumption that disabling a module removes its risk does not hold where code is written as standalone PHP files outside the source tree — another argument for the front controller and source tree conventions.

Release Status and Security Backlog

- **A substantial backlog of reported vulnerabilities remains, with critical items being worked first.** Progress has been slower than hoped, including with AI assistance.
- **The bottleneck is validation rather than the fix itself.** Producing a fix is quick; verifying it against the full test suite adds roughly an hour per fix, which does not scale across a backlog this size. Workflow improvements are being explored.
- **Fax/SMS fixes are next.** The fixes will close the underlying issue, but test credentials from the relevant providers are needed to confirm functionality is preserved. Moving these integrations into their own module is the longer-term answer.
- Broader view carried into the discussion: there are root-level fixes that would be wide-reaching, but the ad hoc structure of parts of the codebase makes clean fixes difficult. Continued progress on the front controller, migration away from ad hoc SQL, and session handling cleanup all reduce the recurrence rate.
- **Release timing:** a cut was planned for 8/5 with the official release the following week, though a slip is likely — critical and high severity items should land first if this is to be a security release.

- **The release call was moved to next week** to get an update on timing and where the group can contribute.
- Mitigating factor: the improved release automation means releases can be run repeatedly, so a slip carries less cost than it once did.

Community Experiment: OpenEMR Modernization via GAI

- **Forum thread:**

  https://community.open-emr.org/t/openemr-modernization-via-gai-experiment-discussion-on-releasing-this-to-public/26923/4
- **Neil Kimber used generative AI to rebuild OpenEMR's feature set on a modern stack in roughly nine days**, reporting approximately 95% feature parity, and has posted to the forum to discuss releasing it publicly.
- The stack replaces PHP, Apache, MariaDB, and AngularJS with React, TypeScript, ASP.NET Core, PostgreSQL, and Node.js.
- It does not reuse OpenEMR code, layouts, or other components, so it is a separate application with a comparable feature set rather than a port of the existing project.
- **The point raised as most meaningful for the community:** whether an existing OpenEMR database can be migrated into it and run from that data. That is the hard part — the OpenEMR schema is strongly denormalized in places, avoids foreign keys, relies on application-level behavior, and mixes design patterns accumulated over the project's history, including Oracle sequences from its earliest days. Any tooling targeting that schema faces the same challenge.

Action Items

Owner	Action
Stephen Waite	Finish PR 13374 to record the OAuth client ID in the API audit log, and add the ability to filter the audit log by client ID in the interface.
Stephen Waite	Post a reply to the AI integration forum thread using the PR as a concrete starting point.
Stephen Nielson	Add a fuller response to the forum thread on direction — delegation vs. impersonation, token exchange, and the identity provider question.
Stephen Nielson	Verify whether the client ID is already being populated into the user ID field in the API response logger listener, and whether that applies only to system clients.
Stephen Nielson	Complete the fax/SMS security fixes. Test credentials are needed from the relevant providers to confirm functionality is preserved.
Stephen Nielson	Evaluate a user/client tagging feature so AI clients can be tagged and audit logs filtered by tag.
Eric Stern	Share an example agent hook configuration that enforces safer command patterns.
Team	Draft a short blog article on vulnerability severity assessment, to reference in future disclosure discussions.
Team	Decide on removal vs. archived-module path for backup.php and other high-risk, low-usage features.
Next week	Release call rescheduled — get an update from Brady on timing and how the group can help clear the security backlog.

Recurring Theme

- Across every topic — the advisory backlog, the forum response, backup.php, and AI access controls — the constraint was maintainer bandwidth rather than technical direction. Community members willing to take on maintenance of specific features or subsystems would have an outsized impact right now.