Andree Salvo

Southern New Hampshire University

CYB 240 – 13711

7-2 Project Two Submission: Recommendations Report

Instructor: Brian Remson

Project Two Scenario One

1. Development Issue/Vulnerability One

Potential area of concern

The development team's current design specifies that web pages display sensitive patient information in clear text using HTTP protocols, and transactions to the web server are unencrypted. This directly aligns with OWASP A02:2021 – Cryptographic Failures, which identifies weaknesses in protecting data during transmission and storage. Because the application will process and display protected health information (PHI), any lack of encryption can expose the organization to serious risks, including data interception through man-in-the-middle (MITM) attacks, HIPAA compliance violations, and possible legal or financial penalties. In healthcare, safeguarding PHI is critical, and failing to implement encryption undermines both patient privacy and organizational trust.

Recommendations

To keep patient data safe, the **"Defense in Depth"** principle should be used. This means protecting it in multiple ways, so if one layer fails, the others keep it secure. All communication between the database, transaction server, and web server should use **TLS 1.3** for encryption in transit. Data stored in the MySQL database should be encrypted with **AES-256** and have secure key management. On top of that, enforce HTTPS only, set up **HSTS** so browsers always use a

secure connection, and monitor for any attempts to downgrade encryption. These combined layers make it significantly more difficult for attackers to intercept or read sensitive information.

Justification

Defense in Depth makes sense here because it doesn't put all the trust in one security measure to protect patient data. If something like HTTPS is bypassed or misconfigured, other layers, like database encryption and strong key management, remain in place to keep the data safe. Stacking these protections makes it significantly harder for attackers to get through, reduces the risk of PHI exposure, and keeps the app compliant with HIPAA and other privacy requirements.

2. Development Issue/Vulnerability One

Potential area of concern - (OWASP A01:2021 – Broken Access Control)

Looking at my chosen scenario, the log-on to the web server is handled through client-side scripting, which makes it a big problem. Client-side code can be viewed, changed, or entirely bypassed by anyone with access to a browser's developer tools. This means that an attacker could skip the login process or change the code to give themselves more access than they're supposed to have. Since this system connects to patient information, this could lead to unauthorized access, PHI exposure, and violations of HIPAA.

The problem of using client-side scripting for logins can be solved if all authentication and authorization are handled on the server rather than relying on the browser. Server-side solutions not only authenticate the user but also perform **role-based access control** to ensure users can only access specific data or features they're approved for.

The authentication system issues secure, short-lived session tokens that are stored as cookies in the browser and protected with HttpOnly, Secure, and SameSite flags. The system also securely

rotates the tokens at login time and destroys them at logout time afterwards. The entire goal for the system’s main principles should follow the Complete Mediation principle, ensuring that the server always controls user access. Because even with a strong browser, an attacker can't talk to the server unless they have been granted special rights to do so.

3. Value of a Security Practitioner in the SDLC

Involving a security practitioner from the very start of the software development life cycle makes a huge difference. Risks can be identified in the planning and design phases by someone who understands the fundamental security design principles. Identifying risks early means addressing problems like unencrypted data transfers or weak access control right away, instead of finding them later when they're harder and more expensive to fix. Risks can be identified not only in the planning phase but also in the design phase, where security controls can and should be built into each phase of the SDLC that we're working on. Failure to build security during planning and design means our only real option in most cases is to find problems using inadequate security design principles during testing.