BẢNG THEO DÕI SỬA ĐỔI

STT	Trang	Nội dung sửa đổi	Ngày có hiệu lực
		Tạo mới	

I. Mục đích

- Đưa ra các tiêu chuẩn để đảm bảo việc xây dựng và triển khai ứng dụng web an toàn.

II. Đối tượng áp dụng

- Tiêu chuẩn này áp dụng với các đơn vị hạch toán phụ thuộc Tập đoàn và các công ty hạch toán độc lập do Tập đoàn sở hữu từ 50% vốn điều lệ trở lên.
- Trong vòng 60 ngày kể từ ngày tiêu chuẩn này có hiệu lực, các đơn vị có trách nhiệm ban hành các hướng dẫn cụ thể cho các framework sử dụng tại đơn vị nhưng phải đảm bảo tuân thủ Tiêu chuẩn này.

Sau khi ban hành, các đơn vị nêu trên phải gửi một bản về Phòng Công nghệ thông tin Tập đoàn để theo dõi, quản lý.

III. Phạm vi áp dụng

- Tiêu chuẩn này áp dụng chung cho tất ứng dụng web được phát triển và sử dụng trong Tập đoàn.

IV. Các yêu cầu

1. Quản lý xác thực

- ***Thông tin định danh***

1.1. Tên đăng nhập phải là duy nhất, chỉ nên chứa tập các ký tự là chữ cái, chữ số, dấu gạch dưới.

1.2. Thiết lập chính sách mật khẩu mạnh:

- Mật khẩu có độ dài tối thiểu là 8 kí tự.
- Chứa chữ cái, chữ số và kí tự đặc biệt.
- Thiết lập blacklist các mật khẩu yếu (VD: 123456A@, 123456a@,…).

1.3. Thiết lập thời gian hết hiệu lực cho mật khẩu tối đa 90 ngày, mật khẩu mới không được trùng với mật khẩu hiện tại.

1.4. Đối với chức năng reset/ quên mật khẩu:

- Đường dẫn reset/quên mật khẩu được gửi qua email phải bị mất hiệu lực sau lần truy cập đầu tiên hoặc sau 8 giờ nếu không được truy cập.
- Nếu chức năng reset/quên mật khẩu thực hiện gửi mật khẩu qua email thì mật khẩu phải được sinh ngẫu nhiên và phải tuân theo chính sách mật khẩu mạnh tại mục 2.

1.5. Chỉ lưu dạng mã hash của mật khẩu trong DB, sử dụng thuật toán hash từ SHA-256 trở lên, thêm chuỗi salt ngẫu nhiên vào mật khẩu trước khi thực hiện hash.

- ***Xử lý xác thực***

1.6. Trả về thông báo chung cho trường hợp người dùng đăng ký thông tin định danh (username, email,…) đã tồn tại tại chức năng đăng ký, hoặc gửi sai thông tin định danh tại các chức năng đăng nhập, reset/quên mật khẩu, đổi địa chỉ email,…

1.7. Bật cơ chế bảo vệ bằng Captcha hoặc các hình thức tương đương khi đăng nhập sai quá 5 lần liên tiếp. Cần triển khai cơ chế này tại các chức năng quan trọng khác của ứng dụng.

1.8. Chỉ sử dụng phương thức POST để submit thông tin định danh, khuyến nghị sử dụng HTTPS cho đường truyền để tăng tính bảo mật.

2. Quản lý phiên đăng nhập

2.1. Session phải được quản lý bởi server, sinh ngẫu nhiên và độ dài tối thiểu là 128-bit.

2.2. Session phải được thiết lập thời gian timeout, giá trị timeout cần cân bằng giữa nhu cầu thương mại và yếu tố bảo mật.

2.3. Tạo mới session sau khi đăng nhập thành công.

2.4. Xóa giá trị sessionid và các dữ liệu gắn với session đó khi người dùng đăng xuất.

2.5. Cấu hình thuộc tính "Secure" đối với các ứng dụng sử dụng HTTPS và "HTTP-Only" cho trường session cookie.

2.6. Đối với các chức năng quan trọng có tương tác với database, ứng với mỗi phiên cần sinh thêm 1 token ngẫu nhiên, và thực hiện kiểm tra tính hợp lệ của token này trước khi xử lý truy vấn từ người dùng.

3. Phân quyền

3.1. Kiểm tra phân quyền dựa trên các đối tượng được lưu tại server (VD: tham số lưu trên session server, dữ liệu lưu trên DB,…).

3.2. Phân quyền tối thiểu, chỉ đáp ứng đủ chức năng và tài nguyên cho người dùng/ứng dụng.

3.3. Phía giao diện người dùng: Chỉ hiển thị các thành phần giao diện, đường dẫn, hàm,... tương ứng với quyền của người dùng.

3.4. Phía server: Kiểm tra quyền tác động của người dùng/ứng dụng trên các hàm và tài nguyên tương ứng trước khi thực hiện bất cứ tác vụ nào tới hệ thống.

3.5. Phải có tính năng xóa phiên làm việc hiện tại của người dùng hoặc các cơ chế tương đương đối với các trường hợp quyền người dùng bị thay đổi hoặc bị disable bởi người dùng có thẩm quyền.

3.6. Không đặt trang quản trị public internet, trong trường hợp bắt buộc phải đặt public cần giới hạn các IP được phép truy cập hoặc sử dụng cơ chế xác thực đa nhân tố (multiple authentications).

4. Tương tác với back-end

- Mã hóa các dữ liệu nhạy cảm trước khi lưu trữ (thông tin tài khoản ngân hàng, private key,…)
- ***SQL***

4.1. Sử dụng mô hình truy vấn prepared statement (parameterized query) hoặc

các hình thức tương đương.

4.2. Trong 1 số trường hợp không sử dụng được các mô hình ở trên, cần thiết lập danh sách whitelist các đầu vào mong muốn.

- ***NoSQL***

4.1. Không mở service ra ngoài public internet, cài đặt trong môi trường mạng an toàn (trusted enviroments).

4.2. Đối với các hệ NoSQL có hỗ trợ xác thực, cần cấu hình xác thực khi truy cập.

4.3. Phụ thuộc vào hệ NoSQL sử dụng, sử dụng các api hỗ trợ truy vấn an toàn hoặc thực hiện escape các ký tự đặc biệt khi xây dựng câu truy vấn.

- ***XPath***

4.1. Thiết lập danh sách whitelist các ký tự đầu vào mong muốn, đầu vào nên là tập hợp của chữ cái, chữ số.

4.2. Lập danh sách blacklist các ký tự đặc biệt (( ) = ‘ [ ] : , * / và dấu cách), loại bỏ các đầu vào có chứa các ký tự nằm trong blacklist.

- ***LDAP***

4.1. Thiết lập danh sách whitelist các ký tự đầu vào mong muốn, đầu vào nên là tập hợp của chữ cái, chữ số.

4.2. Lập danh sách blacklist các ký tự đặc biệt (( ) ; , * | & = và nullbyte), loại bỏ các đầu vào có chứa các ký tự nằm trong blacklist.

- ***Tương tác với OS***

4.1. Sử dụng các API hỗ trợ việc thực thi câu lệnh hệ thống.

4.2. Không truyền trực tiếp dữ liệu người dùng truyền lên tới OS, trong trường hợp bắt buộc cần thiết lập danh sách whitelist các đầu vào mong muốn.

- ***Tương tác với file***

4.1. Không truyền trực tiếp dữ liệu từ người dùng đến các hàm include file.

4.2. Lập danh sách whitelist các định dạng file được phép upload.

4.3. Validate file hợp lệ bằng cách kiểm tra đồng thời file header và phần mở rộng của file.

4.4. Với các trường hợp không bắt buộc thì không lưu file upload trong thư mục web, bỏ quyền thực thi trên thư mục upload.

4.5. Khi cần refer tới các file tồn tại trên hệ thống cần thiết lập danh sách whitelist đầu vào mong muốn hoặc gán các giá trị định danh tương ứng cho các file thay vì truyền tên file.

4.6. Không trả về đường dẫn tuyệt đối của file.

4.7. Tất cả dữ liệu, tài nguyên hệ thống (báo cáo, file upload, file cấu hình...) không được lưu trong thư mục cho phép truy cập trực tiếp không qua xác thực.

- ***Xử lý back-end HTTP request***

4.1. Khi tạo http request phía server, các tham số GET, POST cho request đó tránh tạo từ dữ liệu phía người dùng, hoặc phải được kiểm tra cẩn thận để chống ghi đè các tham số khác.

- **Tương tác với XML**

4.1. Tắt tính năng external entity resolve và remote doctype retrival của xml parser khi đọc dữ liệu xml.

4.2. Kiểm tra dữ liệu người dùng, encode các kí tự đặc biệt (< > ’ ”) khi tạo dữ liệu xml.

5. Kiểm soát dữ liệu đầu vào

5.1. Việc kiểm tra dữ liệu đầu vào phải được thực hiện phía server.

5.2. Thực hiện việc kiểm tra dữ liệu từ tất cả các nguồn dữ liệu có tương tác với người dùng (Các tham số lấy từ GET/POST request, HTTP Headers, dữ liệu lấy từ DB, dữ liệu từ file upload,…).

5.3. Xác định rõ chuẩn định dạng encode của dữ liệu đầu vào, thực hiện validate dữ liệu sau khi đã decode đầu vào về 1 định dạng chuẩn và nhất quán.

5.4. Validate kiểu dữ liệu, phạm vi và độ dài dữ liệu và định dạng dữ liệu.

5.5. Nếu dữ liệu đầu vào bắt buộc là các ký tự đặc biệt, cần thiếp lập danh sách whitelist các ký tự đầu vào mong muốn.

6. Kiểm soát dữ liệu đầu ra

6.1. Phải chỉ rõ character encoding cho dữ liệu đầu ra.

6.2. Response body phải được encode theo ngữ cảnh sử dụng. Một số trường hợp phổ biến:

- Đầu ra là html, thực hiện html encode các kí tự đặc biệt (<>’”&) từ các nguồn dữ liệu không an toàn (Các tham số lấy từ GET/POST request, HTTP Headers, dữ liệu lấy từ DB, dữ liệu từ file upload,… có thể điều khiển được bởi người dùng).
- Đầu ra là json, thực hiện encode dữ liệu trả về dạng object, không trả về dạng mảng.

6.3. Response header: lọc bỏ các kí tự đặc biệt (\n, \r) do dữ liệu người dùng truyền vào.

6.4. Cookie trả về cần giới hạn tối thiểu nhất các thuộc tính (domain, path, httponly, expire, secure). Tránh lưu trữ các dữ liệu nhạy cảm trên cookie, nếu cần lưu trữ các dữ liệu nhạy cảm thì phải thực hiện mã hóa các dữ liệu này với thuật toán đối xứng mạnh và key chỉ được lưu trên server.

6.5. Hạn chế việc chuyển hướng, chuyển tiếp đến các URI khác. Nếu ứng dụng có chức năng này cần phải lập danh sách whitelist các URI được phép thực hiện chuyển hướng, chuyển tiếp.

7. Kiểm soát ngoại lệ và ghi log ứng dụng

7.1. Xử lý các ngoại lệ bằng try-catch và trả về các thông báo lỗi chung đã custom, thông báo lỗi trả về không được chứa các thông tin nhạy cảm của người dùng, hệ thống,...

7.2. Các thông tin lỗi, ngoại lệ này phải được log lại để phục vụ bảo trì, xác định nguyên nhân lỗi ứng dụng.

7.3. File log phải được đặt tại thư mục an toàn ngoài thư mục web.

7.4. Không log lại các dữ liệu nhạy cảm (thông tin người dùng, session id, thông tin hệ thống).

7.5. Giới hạn người dùng cho phép truy cập file log.

8. Sử dụng framework, lib (third-party components)

8.1. Loại các thành phần, lib không cần thiết.

8.2. Sử dụng phiên bản mới nhất của framework tại thời điểm phát triển ứng dụng.

8.3. Thường xuyên cập nhật các bản vá lỗi cho framework.

8.4. Tắt chế độ development của framework khi triển khai ứng dụng thực tế.

9. Xử lý bussiness logic

Xử lý business logic phụ vào từng ứng dụng nhưng lưu ý:

9.1. Lập trình viên phải nắm rõ được toàn bộ luồng nghiệp vụ của ứng dụng, phải xây dựng các cây testcase cho từng nghiệp vụ để tránh bỏ sót các trường hợp có thể xảy ra.

9.2. Các chức năng quan trọng (ví dụ chuyển khoản ngân hàng), sử dụng lock hoặc các hình thức tương đương để tránh lỗi race condition.

PHÒNG CÔNG NGHỆ THÔNG TIN TẬP ĐOÀN	**PHÓ TỔNG GIÁM ĐỐC CHUYÊN TRÁCH**

Nơi nhận:
- Ban TGĐ TĐ (để b/c);
- Như điều II (để t/h);
- Lưu: VT, PCNTT; Dũng 02.