**ÔN TẬP**

**TUẦN 4 – CSRF**

**Câu 1: XSS Stored Attack còn được gọi là:**

**a. Persistent XSS**

b. Reflected XSS

c. DOM-based XSS

d. Non-Persistent XSS

**Câu 2: Biện pháp nào sau đây không giúp ngăn chặn tấn công XSS?**

**a. Sử dụng phương thức GET thay vì POST**

b. Kiểm tra và làm sạch đầu vào người dùng

c. Mã hóa HTML

d. Sử dụng CSP (Content Security Policy)

**Câu 3: Reflected XSS là:**

**a. Script độc được phản hồi ngay khi người dùng truy cập URL**

b. Script độc được lưu trong cơ sở dữ liệu

c. Script độc được chèn vào mã HTML của trang web

d. Script độc chỉ thực thi trên trình duyệt di động

**Câu 4: Ghép cặp các loại tấn công XSS với mô tả tương ứng.**

**a. DOM-based XSS → Thực hiện thay đổi trực tiếp vào cấu trúc DOM của trang, Stored XSS → Script độc được lưu trữ trên server và thực thi khi người dùng truy cập, Reflected XSS → Script độc được gửi cùng URL và phản hồi ngay lập tức**

b. DOM-based XSS → Thực hiện thay đổi trực tiếp vào cấu trúc DOM của trang, Stored XSS → Script độc được lưu trữ trên client và thực thi khi người dùng truy cập, Reflected XSS → Script độc được gửi cùng URL và phản hồi ngay lập tức

c. DOM-based XSS → Thực hiện thay đổi gián tiếp vào cấu trúc DOM của trang, Stored XSS → Script độc được lưu trữ trên server và thực thi khi người dùng truy cập, Reflected XSS → Script độc được gửi cùng URL và phản hồi ngay lập tức

d. DOM-based XSS → Thực hiện thay đổi trực tiếp vào cấu trúc DOM của trang, Stored XSS → Script độc được lưu trữ trên server và thực thi khi người dùng truy cập, Reflected XSS → Script độc được gửi cùng URL và phản hồi sau đó một khoảng thời gian.

**Câu 5: Phương thức HTTP nào không dễ dàng bị lợi dụng trong CSRF?**

**a. PUT**

b. POST

c. GET

d. DELETE

**Câu 6: Reflected XSS dễ bị khai thác khi nào?**

**a. Khi trang web không kiểm tra và làm sạch đầu vào người dùng**

b. Khi trang web chỉ sử dụng phương thức POST

c. Khi trang web không sử dụng giao thức HTTPS

d. Khi người dùng sử dụng trình duyệt cũ

**Câu 7: XSS là viết tắt của:**

**a. Cross-Site Scripting**

b. Cross-Site Request Forgery

c. Client-Side Scripting

d. Cross-Site Reflected Attack

**Câu 8: CSRF Token hoạt động như thế nào?**

**a. Một token duy nhất được tạo ra cho mỗi phiên và gắn vào yêu cầu từ người dùng**

b. Token được lưu trong cookie và tự động gửi kèm theo mọi yêu cầu

c. Token được gửi lại từ server sau mỗi yêu cầu thành công

d. Token được chia sẻ giữa nhiều người dùng cùng lúc

**Câu 9: CSRF là viết tắt của:**

**a. Cross-Site Request Forgery**

b. Cross-Site Script Forgery

c. Client-Side Request Forgery

d. Cross-Site Reflected Forgery

**Câu 10: Biện pháp nào giúp chống lại cả tấn công CSRF và XSS?**

**a. Sanitize đầu vào người dùng**

b. Sử dụng HTTPS

c. Giới hạn thời gian phiên làm việc

d. Yêu cầu người dùng xác thực lại mật khẩu

**Câu 11: Ghép cặp các biện pháp ngăn chặn XSS với mô tả tương ứng.**

**a. Mã hóa HTML → Thay thế các ký tự đặc biệt để tránh thực thi script, Content Security Policy (CSP) → Chặn các script không mong muốn từ việc thực thi, Sanitize đầu vào → Kiểm tra và làm sạch dữ liệu trước khi xử lý**

b. Mã hóa HTML → Thay thế các ký tự đặc biệt để tránh thực thi script, Content Security Policy (CSP) → Chặn các script không mong muốn từ việc thực thi, Sanitize đầu vào → Kiểm tra và làm sạch dữ liệu sau khi xử lý

c. Mã hóa AES → Thay thế các ký tự đặc biệt để tránh thực thi script, Content Security Policy (CSP) → Chặn các script không mong muốn từ việc thực thi, Sanitize đầu vào → Kiểm tra và làm sạch dữ liệu trước khi xử lý

d. Mã hóa HTML → Thay thế các ký tự đặc biệt để tránh thực thi script, Content Security Policy (CSP) → Cho phép các script không mong muốn thực thi, Sanitize đầu vào → Kiểm tra và làm sạch dữ liệu trước khi xử lý

**Câu 12: Ghép cặp các phương pháp bảo mật với lợi ích tương ứng.**

**a. CSRF Token → Đảm bảo rằng chỉ các yêu cầu hợp lệ được xử lý, CAPTCHA → Ngăn chặn các cuộc tấn công tự động bằng robot, Xác thực lại → Tăng cường bảo mật bằng cách yêu cầu người dùng xác nhận hành động quan trọng**

b. CFRF Token → Đảm bảo rằng chỉ các yêu cầu hợp lệ được xử lý, CAPTCHA → Ngăn chặn các cuộc tấn công tự động bằng robot, Xác thực lại → Tăng cường bảo mật bằng cách yêu cầu người dùng xác nhận hành động quan trọng

c. CSRF Token → Đảm bảo rằng chỉ các yêu cầu hợp lệ được xử lý, non-CAPTCHA → Ngăn chặn các cuộc tấn công tự động bằng robot, Xác thực lại → Tăng cường bảo mật bằng cách yêu cầu người dùng xác nhận hành động quan trọng

d. CSRF Token → Đảm bảo rằng chỉ các yêu cầu hợp lệ được xử lý, CAPTCHA → Ngăn chặn các cuộc tấn công tự động bằng robot, Xác thực lại → Tăng cường bảo mật bằng cách yêu cầu người dùng bỏ qua các hành động quan trọng

**Câu 13: XSS xảy ra khi nào?**

**a. Khi ứng dụng web chấp nhận dữ liệu không kiểm tra từ người dùng và phản hồi trong HTML**

b. Khi người dùng không đăng nhập vào hệ thống

c. Khi server lưu trữ các đoạn mã độc trong cơ sở dữ liệu

d. Khi yêu cầu GET được gửi tới server

**Câu 14: Ghép cặp các dạng tấn công XSS với mô tả tương ứng.**

**a. Stored XSS → Script được lưu trên server và thực thi khi người dùng truy cập, DOM-based XSS → Script chỉnh sửa DOM trên trình duyệt người dùng, Reflected XSS → Script được thêm vào URL và phản hồi ngay khi truy cập**

b. Stored XSS → Script được lưu trên client và thực thi khi người dùng truy cập, DOM-based XSS → Script chỉnh sửa DOM trên trình duyệt người dùng, Reflected XSS → Script được thêm vào URL và phản hồi ngay khi truy cập

c. Stored XSS → Script được lưu trên server và thực thi khi người dùng truy cập, DOM-based XSS → Script chỉnh sửa DOM trên trình duyệt máy chủ, Reflected XSS → Script được thêm vào URL và phản hồi ngay khi truy cập

d. Stored XSS → Script được lưu trên server và thực thi khi người dùng truy cập, DOM-based XSS → Script chỉnh sửa DOM trên trình duyệt người dùng, Reflected XSS → Script được thêm vào URI và phản hồi ngay khi truy cập

**Câu 15: Khi nào yêu cầu CSRF được gửi lên server?**

**a. Khi người dùng đã xác thực truy cập vào trang web mục tiêu**

b. Khi người dùng không đăng nhập vào hệ thống

c. Khi server nhận được request từ attacker

d. Khi trang web không có bảo mật HTTPS

**Câu 16: Ghép cặp các dạng tấn công CSRF với mô tả tương ứng.**

**a. XMLHTTPRequest → Được thực hiện qua API AJAX, POST Request → Gửi yêu cầu qua form ẩn với JavaScript, GET Request → Được thực hiện thông qua URL chứa lệnh độc**

b. XMLHTTPRequest → Được thực hiện qua API AJAX, PUT Request → Gửi yêu cầu qua form ẩn với JavaScript, GET Request → Được thực hiện thông qua URL chứa lệnh độc

c. XMLHTTPRequest → Được thực hiện qua API AJAX, PULL Request → Gửi yêu cầu qua form ẩn với JavaScript, GET Request → Được thực hiện thông qua URL chứa lệnh độc

d. XMLHTTPRequest → Được thực hiện qua API AJAX, POST Request → Gửi yêu cầu qua form ẩn với JavaScript, GET Request → Được thực hiện thông qua URI chứa lệnh độc

**Câu 17: Biện pháp nào sau đây giúp ngăn chặn XSS?**

**a. Mã hóa HTML và kiểm tra đầu vào người dùng**

b. Sử dụng giao thức HTTPS

c. Giới hạn quyền truy cập

d. Chặn các URL từ bên thứ ba

**Câu 18: Biện pháp nào sau đây giúp giảm thiểu DOM-based XSS?**

**a. Tránh sử dụng các hàm JavaScript như "innerHTML" hoặc "document.write"**

b. Sử dụng giao thức HTTPS để mã hóa truyền tải

c. Sử dụng mật khẩu mạnh

d. Chặn các yêu cầu từ bên thứ ba

**Câu 19: Ghép cặp các biện pháp ngăn chặn tấn công CSRF với mô tả tương ứng.**

**a. Re-Authentication → Yêu cầu người dùng nhập lại mật khẩu để xác thực hành động quan trọng, CSRF Token → Một token duy nhất được gắn vào mọi yêu cầu từ người dùng, CAPTCHA → Bắt người dùng hoàn thành một bài kiểm tra trước khi gửi yêu cầu**

b. Re-Authentication → Yêu cầu người dùng nhập lại mật khẩu để xác thực hành động quan trọng, CFRF Token → Một token duy nhất được gắn vào mọi yêu cầu từ người dùng, CAPTCHA → Bắt người dùng hoàn thành một bài kiểm tra trước khi gửi yêu cầu

c. Delay-Authentication → Yêu cầu người dùng nhập lại mật khẩu 3 lần để xác thực hành động quan trọng, CSRF Token → Một token duy nhất được gắn vào mọi yêu cầu từ người dùng, CAPTCHA → Bắt người dùng hoàn thành một bài kiểm tra trước khi gửi yêu cầu

d. Re-Authentication → Yêu cầu người dùng nhập lại mật khẩu để xác thực hành động quan trọng, CSRF Token → Một token duy nhất được gắn vào mọi yêu cầu từ người dùng, no-CAPTCHA → Bắt người dùng hoàn thành một bài kiểm tra trước khi gửi yêu cầu

**Câu 20: Điều gì khiến tấn công CSRF đặc biệt nguy hiểm?**

**a. Nạn nhân không nhận thức được rằng họ đã gửi yêu cầu**

b. Yêu cầu chỉ có thể được gửi khi người dùng không đăng nhập

c. CSRF chỉ xảy ra trên trình duyệt di động

d. CSRF yêu cầu nạn nhân nhập mật khẩu lại

**Câu 21: DOM-based XSS tấn công vào đâu?**

**a. Thay đổi cấu trúc DOM trên trình duyệt của nạn nhân**

b. Thay đổi dữ liệu trên server

c. Chuyển hướng trình duyệt đến trang web khác

d. Chèn script độc vào dữ liệu người dùng

**Câu 22: Tấn công CSRF nhằm:**

**a. Thay đổi trạng thái trên server**

b. Đánh cắp thông tin cá nhân

c. Thực hiện tấn công DDoS

d. Lấy cắp cookie

**Câu 23: ấn công XSS xảy ra khi:**

**a. Dữ liệu không được kiểm tra, mã hóa, hoặc làm sạch**

b. Website không hỗ trợ HTTPS

c. Người dùng sử dụng trình duyệt cũ

d. Người dùng nhập sai dữ liệu

**Câu 24: Ghép cặp các phương pháp ngăn chặn CSRF với mô tả tương ứng.**

**a. CAPTCHA → Yêu cầu người dùng hoàn thành bài kiểm tra tương tác, Re-Authentication → Người dùng phải nhập lại mật khẩu trước khi gửi yêu cầu, CSRF Token → Token được thêm vào mỗi yêu cầu để xác thực**

b. CAPTCHA → Không yêu cầu người dùng hoàn thành bài kiểm tra tương tác, Re-Authentication → Người dùng phải nhập lại mật khẩu trước khi gửi yêu cầu, CSRF Token → Token được thêm vào mỗi yêu cầu để xác thực

c. CAPTCHA → Yêu cầu người dùng hoàn thành bài kiểm tra tương tác, Re-Authentication → Người dùng phải nhập lại mật khẩu trước khi gửi yêu cầu, CFRF Token → Token được thêm vào mỗi yêu cầu để xác thực

d. CAPTCHA → Yêu cầu người dùng hoàn thành bài kiểm tra tương tác, Re-Authentication → Người dùng phải nhập lại mật khẩu trước khi gửi yêu cầu, CSRS Token → Token được thêm vào mỗi yêu cầu để xác thực

**Câu 25: Điều kiện nào không gây ra tấn công XSS?**

**a. Sử dụng bộ mã hóa HTML để xử lý dữ liệu đầu vào**

b. Chấp nhận dữ liệu từ người dùng mà không kiểm tra

c. Thực thi script trong trình duyệt của người dùng

d. Phản hồi dữ liệu người dùng trong trang HTML

**Câu 26: Biện pháp nào sau đây giúp ngăn chặn CSRF?**

**a. Sử dụng CSRF Token**

b. Sử dụng mã hóa dữ liệu

c. Sử dụng phương thức GET thay cho POST

d. Không lưu trữ session

**Câu 27: CSRF hoạt động như thế nào?**

**a. Kẻ tấn công gửi yêu cầu thay mặt người dùng đã xác thực tới server**

b. Kẻ tấn công thực hiện mã độc trên máy tính người dùng

c. Kẻ tấn công lấy cắp thông tin đăng nhập của người dùng

d. Kẻ tấn công mã hóa dữ liệu của server