**LỖ HỔNG XÁC THỰC**

**Khái niệm:**

Xác thực là quá trình xác định danh tính của người dùng.

Có ba yếu tố xác thực chính:

* Một thứ bạn biết như mật khẩu hay câu trả lời bảo mật. Nó là yếu tố về kiến thức.
* Một thứ bạn có như điện thoại hay mã bảo mật. Nó là yếu tố về sở hữu.
* Một thứ bạn là như sinh trắc học hay hành vi. Nó là yếu tố về kế thừa.

**Điểm khác nhau giữa xác thực và phân quyền?**

Xác thực là quá trình xác định bạn là ai, còn phân quyền xác định bạn được phép làm gì.

**Lỗ hổng xác thực sinh ra như thế nào?**

Đa số lỗ hổng xác thực sinh ra theo hai cách sau:

Những kĩ thuật xác thực yếu vì chúng thất bại trong việc bảo vệ chống lại tấn công dồn dập.

Lỗi logic hoặc lập trình kém trong thực thi cho phép cơ chế xác thực bị bỏ qua dễ dàng.

Trong phát triển web, lỗi logic có thể làm cho trang web xử lí không đúng như mong đợi. Tuy vậy, vì xác thực rất quan trọng với bảo mật, logic xác thực bị lỗi khiến cho trang web gặp nhiều rủi ro bảo mật hơn.

**Những ảnh hưởng của lỗ hổng xác thực:**

Ảnh hưởng của lỗ hổng xác thực rất dữ dội. Khi kẻ tấn công có thể bỏ qua lớp xác thực hoặc tấn công dồn dập để truy cập vào tài khoản của người dùng khác, họ có thể truy suất tất cả dữ liệu và chức năng mà tài khoản đó có. Nếu bọn họ có thể thâm nhập vào tài khoản có đặc quyền cao hơn như quản trị viên, họ có thể toàn quyền kiểm soát ứng dụng và truy cập trong hệ thống nội bộ.

Thậm chí thâm nhập vào tài khoản có đặc quyền thấp cũng cho phép kẻ tấn công truy suất vào dữ liệu mà họ không nên có như thông tin kinh doanh nhạy cảm. Thậm chí tài khoản không có dữ liệu nhạy cảm thì họ vẫn có thể truy suất những trang khá để mở rộng phạm vi tấn công. Thông thường những cuộc tấn công có mức độ nghiêm trọng cao sẽ không xuất phát từ các trang công cộng mà từ trang nội bộ.

**Những lỗ hổng trong cơ chế xác thực:**

Một hệ thống xác thực của trang web thường chứa nhiều cơ chế xác thực khác nhau nơi mà lỗ hổng sinh ra.

**1. Lỗ hổng trong đăng nhập dựa vào mật khẩu:**

Đối với những trang web xử lí đăng nhập dựa trên mật khẩu, người dùng tự đăng kí tài khoản hoặc được gửi tài khoản bởi quản trị viên. Tài khoản đó có tên người dùng duy nhất và 1 mật khẩu bí mật. Người dùng nhập vào biểu mẫu đăng nhập để xác thực.

Trong viễn cảnh đó, thực tế cho thấy bọn họ chỉ cần biết mật khẩu bí mật là đủ để xác minh danh tính người dùng. Tóm lại, trang web sẽ bị thâm nhập nếu kẻ tấn công có thể thu thập hoặc đoán thông tin đăng nhập của người dùng.

Nó có thể được làm bằng những cách sau.

**Brute-force attacks**

Tấn công dồn dập là khi kẻ tấn công dùng 1 hệ thống thử và sai để cố gắng đoán thông tin đăng nhập hợp lệ. Những cuộc tấn công này được tự động hóa bằng việc sử dụng danh sách tên người dùng và mật khẩu. Tự động hóa quá trình này bằng việc dùng công cụ chuyên dụng cho phép kẻ tấn công đăng nhập nhiều lần với tốc độ cao.

Tấn công dồn dập không phải lúc nào cũng đoán ngẫu nhiên tên người dùng và mật khẩu. Bằng việc sử dụng tính logic và kiến thức của mình, kẻ tấn công có thể tinh chỉnh những cuộc tấn công vũ bão để gia tăng hiệu suất của cuộc tấn công. Những trang web đăng nhập dựa trên mật khẩu có khả năng bị lỗ hổng rất cao nếu họ không thực thi đầy đủ biện pháp để chống lại brute-force.

**Brute-force tên người dùng:**

Tên người dùng rất dễ đoán nếu chúng dựa trên 1 mô hình nhất định như địa chỉ email. Ví dụ, rất phổ biến khi thấy thông tin đăng nhập của doanh nghiệp theo định dạng sau: firstname.lastname@somecompany.com . Tuy nhiên, thâm chí nếu không có định dạng rõ ràng, những tài khoản có đặc quyền cao thường có những tên rất dễ đoán như admin hay administrator .

Trong quá trình rà soát, kiểm tra xem trang web có tiết lộ những tên người dùng ra công chúng không. Ví dụ, bạn có thể truy suất hồ sơ người dùng mà không cần đăng nhập? Thậm chí nếu nội dung của hồ sơ bị ẩn, tên được dùng trong hồ sơ thường giống với tên đăng nhâp. Bạn cũng nên kiểm tra phản hồi HTTP xem có địa chỉ email nào được tiết lộ không. Rất hiếm, những phản hồi chứa địa chỉ email là của người dùng có đặc quyền cao như quản trị hay nhóm hỗ trợ công nghệ.

**Brute-force mật khẩu:**

Mật khẩu cũng có thể bị tấn công dồn dập, với độ khó phụ thuộc vào độ mạnh của mật khẩu. Rất nhiều trang web dùng một vài chính sách về mật khẩu, bắt người dùng tạo mật khẩu đủ mạnh, rất khó để bẻ gãy.

Việc thực thi mật khẩu với:

* Số lượng kí tự tối thiểu
* Sự kết hợp những chữ cái in hoa, in thường
* Ít nhất 1 kí tự đặc biệt

Mật khẩu có nhiều sự hỗn loạn rất khó để máy tính tự bẻ gãy, chúng ta có thể dùng kiến thức cơ bản về hành vi con người để khai thác lỗ hổng mà người dùng vô tình mắc phải. Thay vì tạo một mật khẩu mạnh với sự kết hợp ngẫu nhiên các kí tự, người dùng thường lấy mật khẩu dễ nhớ và cố gắng vặn vẹo nó để phù hợp với chính sách về mật khẩu. Ví dụ, nếu mypassword không được phép, người dùng có thể thử một vài thứ như Mypassword1! hay Myp4$$w0rd.

Trong trường hợp chính sách yêu cầu người dùng thay đổi mật khẩu thường xuyên, họ thường thay đổi theo hướng dễ đoán.Ví dụ, Mypassword1! sẽ trở thành Mypassword1? hay Mypassword2!

Kiến thức về thông tin đăng nhập hay định dạng dễ đoán có thể giúp tấn công dồn dập phức tạp nhưng hiệu quả hơn thay vì chỉ đơn giản lặp lại những mật khẩu yếu.

**Liệt kê tên người dùng:**

Liệt kê tên người dùng là khi kẻ tấn công nhận ra những thay đổi trong hành vi của ứng dụng web để xác định tên người dùng nào là hợp lệ.

Liệt kê tên người dùng thường xảy ra ở trang đăng nhập, ví dụ khi bạn nhập tên người dùng hợp lệ nhưng mật khẩu sai hoặc ở biểu mẫu đăng kí khi bạn nhập tên người dùng đã tồn tại. Nó giúp giảm thời gian và công sức cần thiết để đăng nhập bằng tấn công dồn dập vì kẻ tấn công có thể nhanh chóng tạo ra 1 danh sách ngắn những tên người dùng hợp lệ.

Trong khi cố gắng tấn công dồn dập trang đăng nhập, bạn nên chú ý đến những điểm khác nhau sau:

Status code: trong quá trình tấn công dồn dập, http status code trả về đa số là giống nhau vì các lần đoán đều sai. Nếu 1 lần đoán trả về status code khác thì chắc chắc tên người dùng đó đã đúng. Cách làm tốt nhất cho những trang web là luôn trả về status code giống nhau bất kể đầu ra là gì, nhưng cách làm này thường không được tuân theo.

Thông báo lỗi: thông báo lỗi trả về thường khác nhau phụ thuộc vào tên người dùng và mật khẩu đều sai hay chỉ có mật khẩu là sai. Cách làm tốt nhất đối với những trang web là chỉ trả về những thông báo lỗi chung chung

Thời gian phản hồi: nếu đa số yêu cầu được xử lí với thời gian phản hồi như nhau, bất kì sai lệch từ nó đều chỉ ra rằng một vài thứ khác đang diễn ra đằng sau. Nó chỉ ra rằng tên người dùng có thể đúng. Ví dụ trang web sẽ chỉ kiểm tra mật khẩu đúng nếu tên người dùng hợp lệ. Bước kiểm tra này có thể làm tăng thời gian phản hồi. Nó rất tế nhị nhưng kẻ tấn công có thể làm sự trì hoãn rõ ràng hơn bằng cách nhập mật khẩu dài.

**Bảo vệ brute-force bị lỗi**

Tấn công dồn dập sẽ liên quan đến những lần đoán thất bại trước khi kẻ tấn công có thể thành công thâm nhập tài khoản.

Về mặt logic, bảo vệ brute-force sẽ làm cho quá trình tự động hóa trở nên phức tạp nhất có thể và làm chậm tốc độ mà kẻ tấn công có thể cố gắng đăng nhập.

Có hai cách phổ biến để ngăn chặn những cuộc tấn công dồn dập:

* Chặn tài khoản nếu người dùng đăng nhập thất bại quá nhiều
* Chặn địa chỉ IP nếu người dùng đăng nhập nhiều lần liên tiếp

Cả hai phương pháp đều cung cấp các mức độ bảo vệ khác nhau, nhưng vẫn có lỗ hổng ở phần thực thi.

Ví dụ, bạn thường thấy địa chỉ IP bị chặn nếu bạn đăng nhập thất bại nhiều lần. Trong một số trường hợp, bộ đếm số lần thất bại sẽ đặt lại nếu địa chỉ IP đăng nhập thành công. Kẻ tấn công sẽ phải đăng nhập tài khoản của mình sau mỗi lần thử để ngăn chặn giới hạn này.

**Khóa tài khoản:**

Một cách mà những trang web ngăn chặn tấn công dồn dập là khóa tài khoản nếu có những hành vi đáng ngờ, thường là số lần đăng nhập thất bại.

Chỉ với lỗi đăng nhập bình thường, phản hồi của máy chủ cũng chỉ ra rằng tài khoản đã bị khóa cũng có thể giúp kẻ tấn công liệt kê tên người dùng.

Việc khóa tài khoản chống lại tấn công dồn dập 1 tài khoản nhất định nhưng sẽ thất bại nếu kẻ tấn công chỉ muốn truy cập vào tài khoản bất kỳ.

Phương pháp sau có thể được dùng để qua mặt loại bảo vệ này:

1. Thiết lập 1 danh sách những tên người dùng hợp lệ
2. Tạo 1 danh sách mật khẩu nhỏ. Quan trọng là số mật khẩu bạn chọn không vượt quá số lần đăng nhập cho phép. Ví dụ, nếu giới hạn là 3, bạn chỉ được tối đa 3 lần đoán
3. Sử dụng công cụ như Burp Intruder, thử từng mật khẩu với danh sách người dùng. Như vậy, bạn vẫn có thể tấn công dồn dập 1 tài khoản mà không sợ bị khóa tài khoản.

Việc khóa tài khoản cũng thất bại trước cuộc tấn công nhồi nhét tài khoản. Nó liên quan đến việc dùng 1 danh sách đồ sộ chứa các cặp username:password, bao gồm thông tin đăng nhập bị đánh cắp trong các vụ đánh cắp dữ liệu.

Việc nhồi nhét thông tin đăng nhập dựa trên sự thật là rất nhiều người dùng chung tên và mật khẩu cho nhiều trang web. Việc khóa tài khoản không bảo vệ được trước cuộc tấn công nhồi nhét tk vì mỗi tên người dùng chỉ đăng nhập 1 lần.

**Giới hạn thời gian đăng nhập:**

Một số trang web khác cố gắng ngăn chặn tấn công dồn dập qua user rate limiting. Trong trường hợp đó, đăng nhập quá nhiều lần trong 1 thời gian ngắn làm cho địa chỉ IP của bạn bị chặn.

Địa chỉ IP có thể không bị chặn bằng những cách sau:

* Tự động hóa sau 1 khoảng thời gian nhất định
* Thủ công bởi quản trị viên
* Thủ công bởi người dùng sau khi hoàn thành CAPTCHA

Giới hạn thời gian đăng nhập được khuyến khích dùng hơn khóa tài khoản vì ít bị liệt kê tên người dùng hơn và tránh những cuộc tấn công DoS.

Tuy vậy, nó vẫn không hoàn toàn bảo mật. Khi giới hạn dựa trên rate của những yêu cầu HTTP từ địa chỉ IP của người dùng, nó vẫn có thể bị bỏ qua nếu bạn có thể đoán nhiều mật khẩu trong 1 yêu cầu.

**HTTP basic authentication:**

Mặc dù khá cũ, nhưng tính đơn giản và dễ thực thi nên đôi khi bạn vẫn thấy xác thực http basic được dùng.

Máy khách nhận 1 mã xác thực từ máy chủ, được nối từ tên người dùng và mật khẩu và mã hóa Base64. Mã này được lưu trữ và quản lí bởi trình duyệt, sẽ tự động thêm vào header Authorization ở mỗi yêu cầu tiếp theo như sau:

Authorization: Basic base64(username:password)

Với một vài lí do, nó không được xem là phương pháp xác thực an toàn. Thứ nhất, nó gửi thông tin đăng nhập của người dùng ở mỗi yêu cầu. Nếu trang web không thực thi HSTS, nó sẽ bị đánh cắp trong 1 cuộc tấn công man-in-the-middle.

Thêm nữa, việc thực thi xác thực http basic không hỗ trợ bảo vệ brute-force. Vì mã vẫn chứa những giá trị tĩnh, nó vẫn bị tấn công dồn dập.

Việc xác thực dựa trên http basic cũng bị dính lỗ hổng liên quan đến phiên, đặc biệt là CSRF.

**2. Những lỗ hổng trong xác thực nhiều yếu tố:**

Trong mục này, chúng ta sẽ tìm kiếm những lỗ hổng có thể xảy ra ở xác thực nhiều yếu tố.

Xác thực sinh trắc học là không thực tế với những trang web. Tuy vậy, ngày càng phổ biến thấy xác thực hai yếu tố dựa trên thứ bạn biết và thứ bạn có. Nó yêu cầu người dùng nhập cả mật khẩu và mã code từ thiết bị của họ.

Xác thực hai yếu tố chỉ bảo mật bởi quá trình thực thi. Nếu thực thi kém cỏi, nó vẫn bị bỏ qua dễ dàng.

Đáng chú ý là lợi ích của xác thực nhiều yếu tố chỉ có khi xác thực những yếu tố khác nhau.

**Mã xác thực hai yếu tố:**

Mã xác thực thường được đọc bởi người dùng từ thiết bị vật lý. Rất nhiều trang web bảo mật bây giờ cung cấp cho người dùng thiết bị chuyên dụng cho mục đích này như mã RSA hay bàn phím để truy cập ngân hàng online. Ngoài mục đích bảo mật, những thiết bị này có thể tạo những mã xác thực trực tiếp. Nó cũng phổ biến cho những ứng dụng điện thoại chuyên dụng như Google Authenticator.

Mặt khác, một vài trang web gửi mã xác thực như tin nhắn văn bản, nó rất dễ bị lạm dụng. Đầu tiên, đoạn mã được gửi qua tin nhắn thay vì được tạo bởi thiết bị. Dễ có rủi ro tráo đổi Sim, nhờ đó kẻ tấn công lấy được thẻ sim với số điện thoại của nạn nhân. Kẻ tấn công sẽ nhận được tất cả tin nhắn gửi đến, bao gồm mã code.

**Bỏ qua xác thực hai yếu tố:**

Rất nhiều lần, xác thực hai yếu tố bị sai xót đến mức chúng bị bỏ qua hoàn toàn.

Nếu người dùng đầu tiên được yêu cầu nhập mật khẩu, sau đó nhập đoạn mã trên 1 trang khác, rõ ràng người dùng đã ở trạng thái đăng nhập trước khi họ nhập mã. Trong trường hợp này, rất đáng để xem xét bạn có thể bỏ qua những trang đó sau bước xác thực đầu tiên. Rất hiếm khi bạn sẽ thấy 1 trang web không kiểm tra xem bạn đã hoàn thành bước thứ hai trước khi tải lại trang.

**Xác thực hai yếu tố bị lỗi logic:**

Xác thực hai yếu tố bị lỗi logic khi người dùng hoàn thành bước đăng nhập ban đầu, nhưng trang web không xác thực người dùng đó có đang hoàn thành bước thứ hai không.

Ví dụ, người dùng đăng nhập ở bước đầu như sau:

POST /login-steps/first HTTP/1.1

Host: vulnerable-website.com

...

username=carlos&password=qwerty

Sau đó họ được gán 1 cookie trước khi được đưa đến bước xác thực thứ hai:

HTTP/1.1 200 OK

Set-Cookie: account=carlos

GET /login-steps/second HTTP/1.1

Cookie: account=carlos

Khi gửi mã code, yêu cầu dùng cookie đó để xác thực xem tài khoản nào đang truy cập

POST /login-steps/second HTTP/1.1

Host: vulnerable-website.com

Cookie: account=carlos

...

verification-code=123456

Trong trường hợp này, kẻ tấn công có thể đăng nhập bằng thông tin đăng nhập của mình nhưng đến bước thứ hai họ thay đổi giá trị của cookie là tên người dùng vô danh

POST /login-steps/second HTTP/1.1

Host: vulnerable-website.com

Cookie: account=victim-user

...

verification-code=123456

Điều này rất nguy hiểm nếu kẻ tấn công sau đó có thể tấn công dồn dập mã xác thực vì nó sẽ cho phép họ đăng nhập tài khoản người dùng vô danh mà không cần mật khẩu.

**Tấn công dồn dập mã xác thực 2FA:**

Cùng với mật khẩu, trang web cần ngăn chặn tấn công dồn dập mã xác thực 2FA. Điều này rất quan trọng vì mã code thường chỉ có 4 hay 6 số. Không có lớp bảo vệ, việc bẻ gãy mã code như vậy rất dễ dàng.

Một vài trang web ngăn chặn điều này bằng cách tự động đăng xuất nếu họ nhập sai mã code nhiều lần. Nó không hiệu quả vì kẻ tấn công có thể tự động quá trình nhiều bước này bằng Burp macro.

**3. Lỗ hổng trong những cơ chế xác thực khác**

Ngoài chức năng đăng nhập cơ bản, đa số trang web cung cấp thêm chức năng cho phép người dùng quản lí tài khoản. Ví dụ, người dùng có thể thay đổi hoặc cập nhật lại mật khẩu khi họ quên. Các cơ chế này có thể bị khai thác bởi kẻ tấn công.

Trang web thường bảo vệ trang đăng nhập khỏi những lỗ hổng nghiêm trọng. Nhưng thật dễ dàng để bỏ qua thực tế là bạn cần thực hiện các bước tương tự để đảm bảo rằng chức năng liên quan cũng mạnh mẽ như nhau. Điều này đặc biệt quan trọng trong trường hợp kẻ tấn công có thể tạo tài khoản của riêng họ và do đó, có quyền truy cập dễ dàng để nghiên cứu các trang bổ sung này.

**Giữ người dùng ở trạng thái đăng nhập**

Một tính năng phổ biến là giữ trạng thái đăng nhập sau khi đóng phiên của trình duyệt. Nó thường hiện ra 1 hộp có nhãn là “Remember me” hay “Keep me logged in”.

Chức năng này được thực thi bằng cách tạo ra 1 mã “remember me” được lưu trữ trong 1 cookie tạm thời. Sở hữu cookie này sẽ cho phép bạn bỏ qua toàn bộ quá trình đăng nhập, tốt nhất là cookie này không thể đoán được. Tuy nhiên, một vài trang web tạo cookie dựa trên sự kết hợp của các giá trị tĩnh có thể đoán được như tên người dùng, thời gian tạo, thậm chí là mật khẩu. Cách tiếp cận này khá nguy hiểm nếu kẻ tấn công có thể tạo tài khoản của riêng mình vì họ có thể nghiên cứu cookie và suy ra cách chúng được tạo. Khi họ tìm ra được thuật toán, họ có thể thử tấn công dồn dập cookie của người dùng khác để giành quyền truy cập.

Một vài trang web giả định nếu cookie được mã hóa thì sẽ không đoán được thậm chí là dùng những giá trị tĩnh. Điều này có thể đúng nếu được mã hóa tốt, nhưng mã hóa cookie ngây thơ bằng Base64 sẽ không mang lại kết quả gì. Thậm chí mã hóa thích hợp với hàm băm 1 chiều cũng không bảo vệ được. Nếu kẻ tấn công có thể xác định thuật toán băm và salt không được dùng, họ có thể tấn công dồn dập cookie bằng việc hash wordlists. Phương pháp này có thể bỏ qua giới hạn đăng nhập nếu nó không áp dụng cho việc đoán cookie.

Thậm chí nếu kẻ tấn công không thể tạo tài khoản của mình, họ vẫn có thể khai thác lỗ hổng này. Bằng các kỹ thuật thông thường như XSS, kẻ tấn công có thể đánh cắp cookie “remember me” của người dùng và suy ra cách cookie được tạo. Nếu trang web được xây dựng bằng thư viện mã nguồn mở, cách tạo cookie có thể được ghi lại công khai.

Trong một vài trường hợp hiếm khác, có thể quan sát được mật khẩu của người dùng rõ ràng từ cookie, thậm chí nó được hash. Danh sách mật khẩu được hash có rất nhiều, nếu mật khẩu của người dùng có trong danh sách đó, giải mã hash có thể rất đơn giản. Điều đó chứng tỏ tầm quan trọng của salt.

**Cập nhật lại mật khẩu người dùng**

Trong thực tế một vài người sẽ quên mật khẩu. Xác thực dựa trên mật khẩu sẽ không còn khả thi trong diễn cảnh này, trang web phải dựa vào những phương thức khác để đảm bảo người dùng thật đang cập nhật lại mật khẩu. Vì vậy, chức năng cập nhật mật khẩu cũng rất nguy hiểm và cần được thực thi an toàn.

Chức năng được thực thi bằng rất nhiều cách:

**Gửi mật khẩu bằng email:**

Gửi cho người dùng mật khẩu cũ sẽ không khả thi nếu trang web xử lí mật khẩu an toàn. Thay vào đó, một vài trang web tạo 1 mật khẩu mới và gửi nó qua email.

Nói chung, việc gửi những mật khẩu qua những kênh không an toàn thì nên tránh. Trong trường hợp này, bảo mật dựa vào mật khẩu được tạo sẽ hết hạn sau một khoảng thời gian rất ngắn hay người dùng thay đổi lại mật khẩu ngay lập tức. Mặt khác cách tiếp cận này rất dễ bị tấn công bởi cuộc tấn công MITM.

Email cũng thường không được coi là an toàn do hộp thư đến rất bền và không được thiết kế để lưu trữ những thông tin nhạy cảm. Rất nhiều người dùng cũng đồng bộ hóa hộp thư giữa nhiều thiết bị qua những kênh không an toàn.

**Cập nhật lại mật khẩu bằng Url:**

Một phương pháp mạnh mẽ hơn để đặt lại mật khẩu là gửi 1 url duy nhất đến người dùng để đưa họ đến trang cập nhật mật khẩu. Những thực thi thiếu an toàn của phương pháp này là dùng 1 Url có tham số dễ đoán để xác định tài khoản nào được cập nhật, ví dụ:

http://vulnerable-website.com/reset-password?user=victim-user

Trong ví dụ này, kẻ tấn công có thể thay đổi tham số user để đến trang mà họ có thể đặt lại mật khẩu mới cho người dùng vô danh.

Cách tốt hơn là tạo ra 1 mã khó đoán, high-entropy. Url này không nên cung cấp bất cứ manh mối gì về việc cập nhật lại mật khẩu.

http://vulnerable-website.com/reset-password?token=

a0ba0d1cb3b63d13822572fcff1a241895d893f659164d4cc550b421ebdd48a8

Khi người dùng truy cập Url này, hệ thống nên kiểm tra xem mã này có tồn tại ở phía sau hay không và nếu có, mật khẩu của người dùng nào sẽ được đặt lại. Mã này sẽ hết hạn sau 1 khoảng thời gian ngắn và bị hủy ngay lập tức nếu mật khẩu đã được cập nhật.

Tuy nhiên, một số trang web cũng không thể xác thực lại mã thông báo khi biểu mẫu reset được gửi. Trong trường hợp này, kẻ tấn công có thể chỉ cần truy cập biểu mẫu từ tài khoản của chính họ, xóa token và tận dụng trang này để đặt lại mật khẩu cho người dùng vô danh.

**Thay đổi mật khẩu người dùng**

Là nhập mật khẩu cũ và sau đó mật khẩu mới hai lần. Chức năng này về cơ bản giống như trang đăng nhập là kiểm tra tên người dùng và mật khẩu.

Chức năng thay đổi mật khẩu có thể rất nguy hiểm nếu cho phép kẻ tấn công truy cập nó mà không cần đăng nhập. Ví dụ, nếu tên người dùng bị ẩn, kẻ tấn công có thể chỉnh sửa nó trong yêu cầu để nhắm đến người dùng tùy ý. Nó có thể được khai thác để liệt kê tên người dùng và tấn công dồn dập mật khẩu.

**4. Làm thế nào để bảo mật những cơ chế xác thực?**

**Quan tâm đến thông tin đăng nhập người dùng**

Những cơ chế xác thực mạnh mẽ nhất cũng kém hiệu quả nếu bạn vô tình tiết lộ danh sách thông tin đăng nhập cho kẻ tấn công. Bạn không nên gửi dữ liệu đăng nhập qua những kết nối không được mã hóa. Mặc dù bạn đã thực thi HTTPs cho những yêu cầu đăng nhập, đảm bảo rằng bạn thực thi nó bằng cách chuyển hướng những yêu cầu http sang https

Bạn cũng nên đảm bảo không có tên người dùng hay địa chỉ email nào bị tiết lộ qua những hồ sơ công khai hay phản ánh trong những phản hồi http.

**Đừng tin tưởng vào người dùng để bảo mật**

Những phương pháp xác thực nghiêm ngặt thường yêu cầu thêm 1 vài nỗ lực từ người dùng. Nhưng không thể tránh khỏi một vài người sẽ đơn giản cho mình những nỗ lực này. Do đó, bạn cần thực thi

Ví dụ rõ ràng nhất là thực thi chính sách bảo mật hiệu quả. Một vài chính sách cũ đã thất bại vì mọi người đưa mật khẩu dễ dự đoán vào chính sách. Thay vào đó, có thể hiệu quả hơn nếu triển khai một số trình kiểm tra mật khẩu đơn giản, sẽ cho phép người dùng thử mật khẩu và cung cấp phản hồi về độ mạnh trong thời gian thực. Một ví dụ đơn giản là thư viện Javascript zxcvbn, được phát triển bởi Dropbox. Bằng việc chỉ cho phép những mật khẩu mạnh bởi password checker, bạn có thể yêu cầu người dùng nhập mật khẩu mạnh hơn.

**Ngăn chặn việc liệt kê tên người dùng**

Kẻ tấn công sẽ dễ phá vỡ những cơ chế xác thực nếu bạn tiết lộ 1 người dùng đang tồn tại trên hệ thống. Trong một số tình huống, vì bản chất của trang web, việc biết 1 người dùng cụ thể có 1 tài khoản chính là thông tin nhạy cảm.

Bất cả tên người dùng có hợp lệ, điều quan trọng là thông báo lỗi phải giống hệt nhau, chung chung. Bạn nên thường xuyên trả về trạng thái http giống nhau với mỗi yêu cầu đăng nhập và cuối cùng làm cho thời gian phản hồi trong nhiều tình huống khác nhau càng giống nhau càng tốt.

**Chống brute-force nghiêm ngặt**

Tấn công dồn dập dễ như thế nào, do đó bạn phải có nhiều bước để ngăn chặn, ít nhất là làm gián đoạn bất cứ nỗ lực tấn công dồn dập trang đăng nhập.

Một trong những phương pháp hiệu quả hơn là thực thi rate limiting dựa trên IP người dùng. Nó ngăn chặn kẻ tấn công thao túng địa chỉ IP. Bạn nên yêu cầu người dùng hoàn thành 1 bài kiểm tra CAPTCHA với mỗi nỗ lực đăng nhập sau khi giới hạn bị chạm đến.

Hãy nhớ là điều này sẽ không loại bỏ hoàn toàn mối đe dọa từ tấn công dồn dập. Tuy nhiên làm cho quá trình trở nên tẻ nhạt và thủ công nhất có thể để kẻ tấn công từ bỏ và tìm kiếm 1 phần mềm khác.

**Kiểm tra kĩ logic xác minh của bạn**

Rất dễ để có lỗi logic trong code, trong trường hợp là xác thực, có thể thâm nhập hoàn toàn trang web và người dùng. Kiểm tra kĩ logic xác minh để loại bỏ lỗi có thể là chìa khóa cho xác thực mạnh mẽ.

**Đừng quên những chức năng bổ sung**

Không nên chỉ tập trung vào trang đăng nhập và bỏ qua các chức năng khác liên quan đến xác thực. Điều này rất quan trọng với những trường hợp kẻ tấn công được tự do đăng kí tài khoản và khám phá chức năng này. Nhớ rằng cập nhật hay thay đổi mật khẩu cũng là bề mặt tấn công hợp lệ như trang đăng nhập và do đó chúng phải mạnh mẽ như nhau.

**Xác thực nhiều lớp phù hợp**

Trong khi xác thực nhiều yếu tố có thể không thực tế với mọi trang web, khi thực hiện đúng sẽ bảo mật hơn đăng nhập chỉ dựa vào mật khẩu. Nhớ rằng xác thực nhiều trường hợp của 1 yếu tố thì không là xác thực nhiều yếu tố.

2FA dựa vào SMS xác thực hai yếu tố (thứ bạn biết và thứ bạn có). Tuy nhiên, nó có thể bị lạm dụng thông qua hoán đổi sim nên hệ thống này không đáng tin cậy.

Hợp lí hơn, 2FA nên được thực thi bằng 1 thiết bị chuyên dụng tạo ra mã xác thực trực tiếp.

Cuối cùng, phải đảm bảo tính logic của 2FA để nó không bị bỏ qua dễ dàng.