



# 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

---

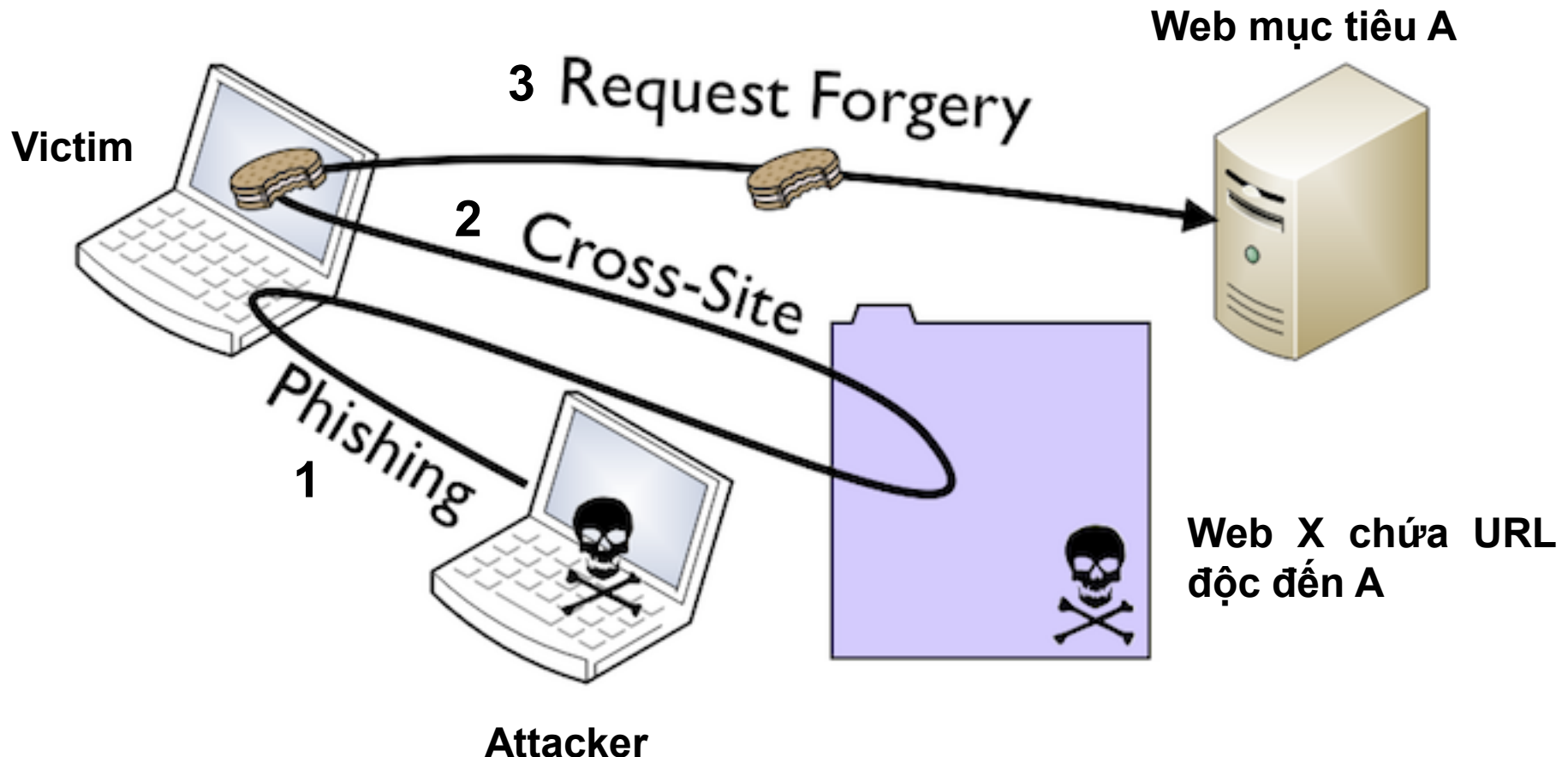
- Tên khác: CSRF, XSRF, one-click attack, Sea Surf, Hostile Linking, session riding
- Tấn công **giả mạo chứng thực** của người dùng để thực hiện các hành động mà kẻ tấn công mong muốn
- **Mục tiêu:** thực hiện yêu cầu thay đổi trạng thái, không nhằm đánh cắp dữ liệu, với sự hỗ trợ của các kỹ thuật xã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

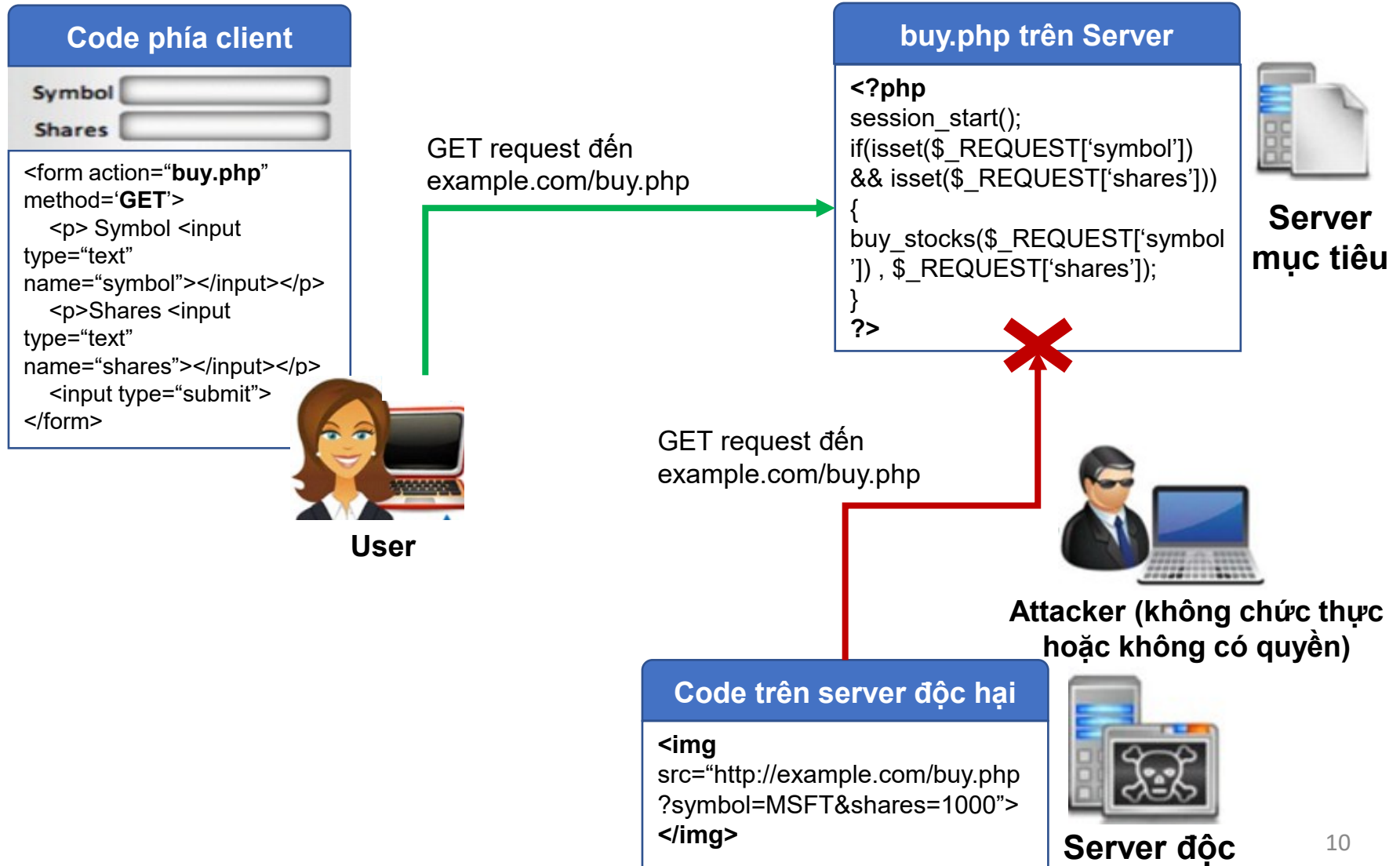
# Nguyên tắc hoạt động của CSRF

---

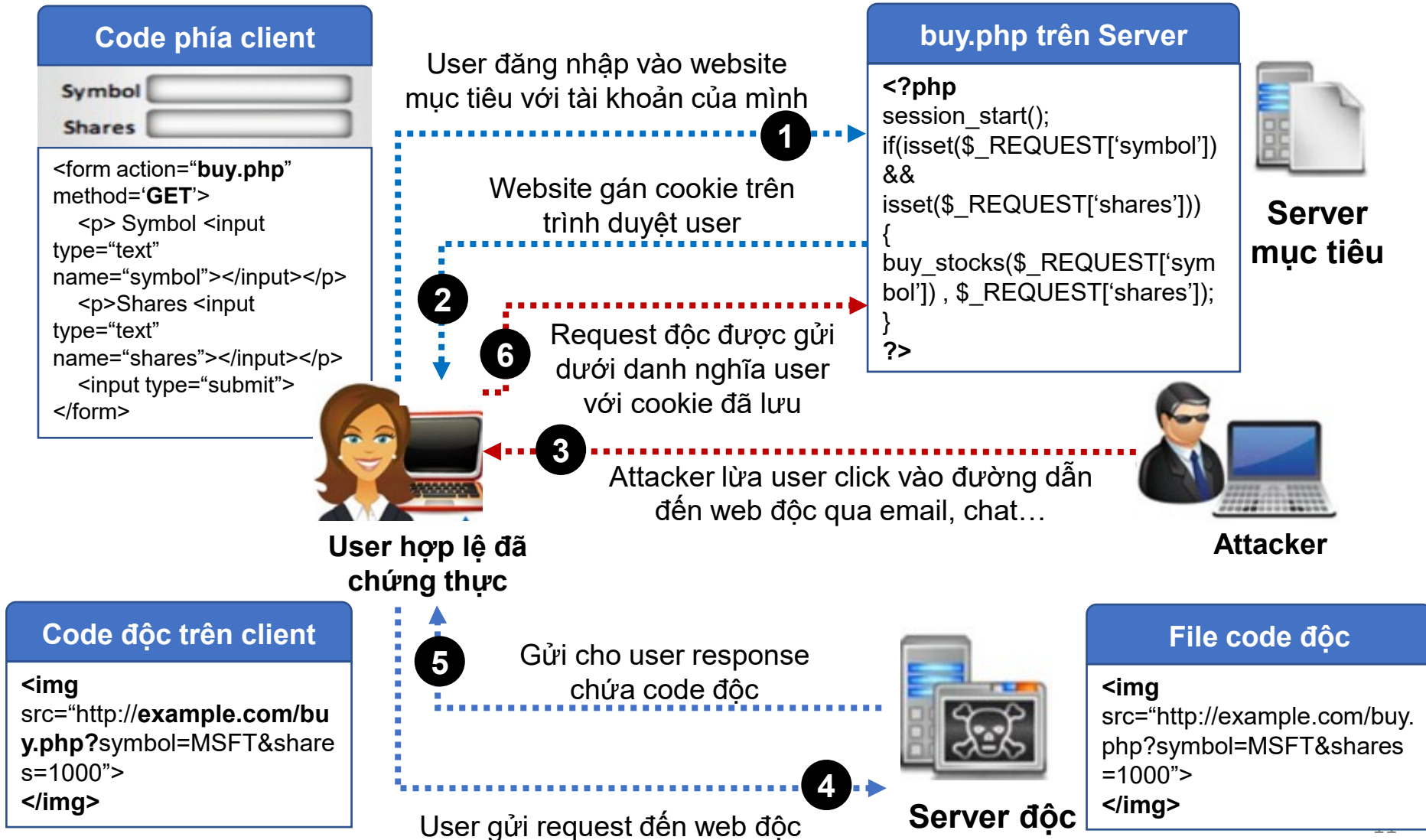
- Khi một request được gửi, trình duyệt sẽ xem có cookie nào thuộc về domain của URL đó để gửi cùng với request. Do nạn nhân đã đăng nhập nên **cookie của nạn nhân sẽ được gửi** lên server và trang **web A** sẽ **nhầm** rằng đây là **request do nạn nhân muốn thực hiện**
- Từ đó, Hacker đã mạo danh nạn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



# CSRF – Ngữ cảnh

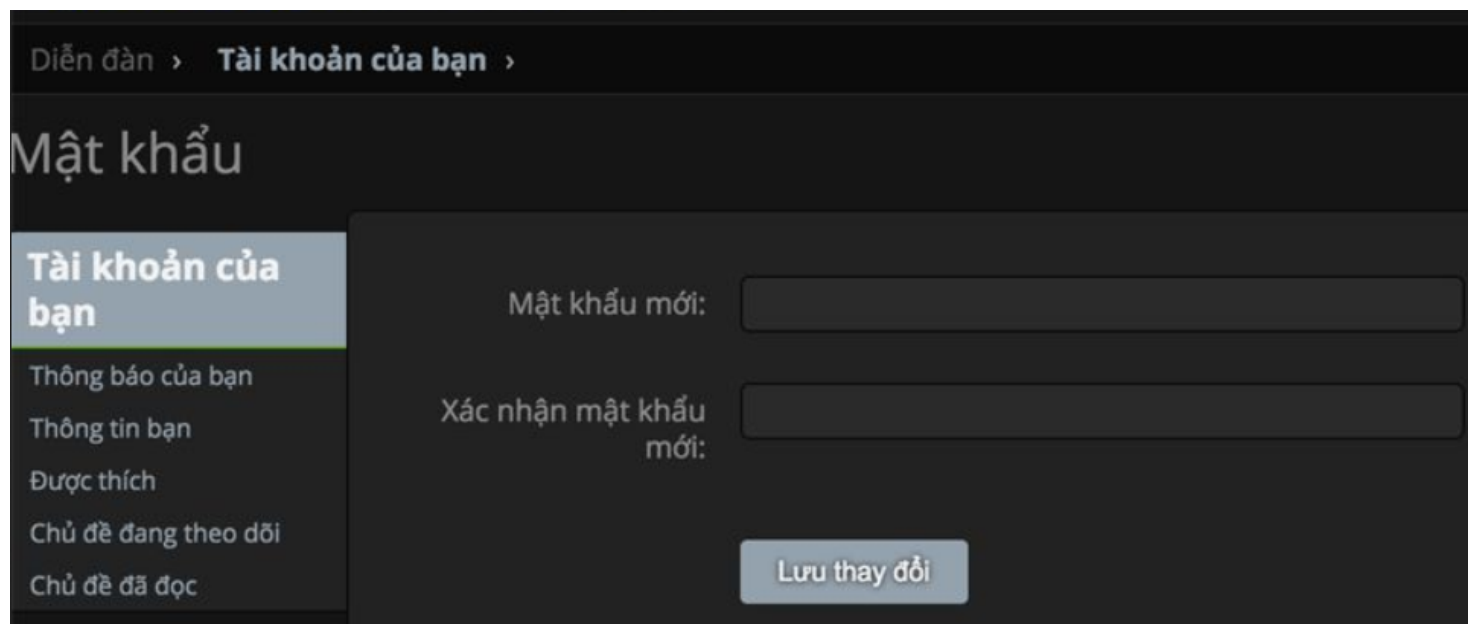


# CSRF – Cơ chế hoạt động



# Cách tấn công – Nội dung web độc

- Dùng thẻ **<img>**, **<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 interface for a user's account. At the top, there is a breadcrumb trail: "Diễn đàn > Tài khoản của bạn >". Below this, the title "Mật khẩu" (Password) is displayed. 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:" (New password) and "Xác nhận mật khẩu mới:" (Confirm new password). Below these fields is a button labeled "Lưu thay đổi" (Save changes).

- 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:

<a

href="http://bank.com/transfer.do?acct=MARIA&amount=100000">View my Pictures!</a>

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 dẫn 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



# 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

# Bảo mật web và ứng dụng



Trường ĐH CNTT TP. HCM