1

Lab

Same-Origin Policy – SOP

Khai thác Web SOP

Thực hành Bảo mật web và ứng dụng

GVTH: Ung Văn Giàu

cuu duong than cong . com

Học kỳ I – Năm học 2017-2018

Lưu hành nội bộ

A. TỔNG QUAN

1. Giới thiệu

Mô hình bảo mật của trình duyệt web dựa trên chính sách cùng nguồn gốc (Same-Origin Policy, viết tắt là SOP). Mô hình này cung cấp một vài đặc trưng bảo vệ cơ bản cho ứng dụng web.

2. Mục tiêu

Giúp sinh viên có được kiến thức về SOP phục vụ cho các bài lab khác như Cross-site Scripting và Cross-site request forgery.

3. Môi trường & mô hình mạng

Sử dụng máy ảo *SEEDUbuntun12.04.zip* để thực hành bài lab. Link tải máy ảo:

http://www.cis.syr.edu/~wedu/SEEDUbuntu12.04.zip hoặc

https://drive.google.com/file/d/0B2xNqn8OtWQ2cWdzMUdlWXhudzQ/view?usp=sharing

a) Cấu hình môi trường

Lab Web SOP cần:

- Trình duyệt Firefox có cài extension LiveHTTPHeaders (có thể sử dụng WireShark để thay thế cho LiveHTTPHeaders)
- Apache web server
- Ứng dụng quản lý dự án web Collabtive

Khởi động Apache Server:

```
sudo service apache2 start
```

Ứng dụng web Collabtive là một hệ thống quản lý dự án trên nền web. Ứng dụng có sẵn một số tài khoản. Tài khoản admin: admin/admin.

b) Cấu hình DNS

Đã cấu hình những URL cần thiết cho lab.

URL	Mô tả	Directory
http://www.soplab.com		/var/www/SOP/
http://www.soplabattacker.com	Attacker	/var/www/SOP/attacker
http://www.soplabcollabtive.com	Collabtive	/var/www/SOP/soplabCollabtive

c) Cấu hình Apache Server

Sử dụng Apache server để host tất cả website sử dụng cho lab.

Tập tin cấu hình có tên *default* trong thư mục *"/etc/apache2/sites-available"*.

Các thông tin cần thiết cho cấu hình:

- *NameVirturalHost* *: web server sử dụng tất cả địa chỉ IP.

- Mỗi website có một khối *VirtualHost* chỉ định URL cho website và đường dẫn thư mục chứa mã nguồn cho website.

d) Tắt chế độ Cache

Lab yêu cầu thực hiện một vài chỉnh sửa ứng dụng web. Do đó, để chắc rằng trình duyệt luôn lấy dữ liệu mới từ web được chỉnh sửa chứ không từ cache.

Để disable cache bạn gõ *about:config* trên thanh địa chỉ của trình duyệt Firefox và thiết lập như sau:

browser.cache.memory.enable	/* thiết lập false, mặc định true*/
browser.cache.disk.enable	/* thiết lập false, mặc định true*/
browser.cache.check_doc_frequency	/* 1 = mỗi lần, mặc định 3 là khi cần*/

B. THỰC HÀNH

1. DOM và Cookies

Mục tiêu: làm quen với DOM APIs, dùng chỉnh sửa cookies và nội dung trang web.

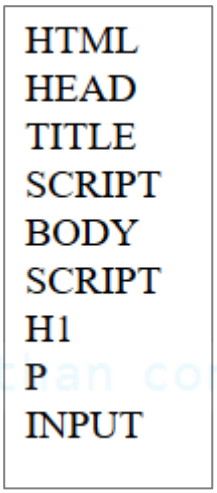
a. Viết hàm JavaScript duyệt và hiển thị cây DOM[2] cho trang web được đính kèm (SOP_DOM_Task_1.html).

Hướng dẫn:
Hình bên dưới giải thích nội dung các hàm và đưa ra hướng dẫn, cách sử dụng cho hàm cần viết tiếp trong file SOP_DOM_Task_1.html. Các bạn sẽ dùng một editor tùy chọn (Sublime Text, Notepad++, Visual Studio Code,...) để mở xem và viết tiếp vào function duyetDom(parent).

```
    <script>
        /* Hàm thêm node <h1> và node <p> vào <body> */
        function appendp() {
            // Tạo node và nội dung cho <h1>
            var h1_node = document.createElement("h1");
            h1_node.innerHTML = "Self-modifying HTML Document";

            // Thêm node <h1> và <body>
            document.childNodes[0].childNodes[2].appendChild(h1_node);

            // Tạo node và nội dung cho <p>
            var p_node = document.createElement("p");
            p_node.innerHTML = "This web page illustrates how DOM API can be used to modify a web page";

            // Thêm node <p> và <body>
            document.childNodes[0].childNodes[2].appendChild(p_node);
        }
        /* Lấy node con của <html> */
        function gethtmlchildren() {
            // Lấy node <html>
            var entiredoc = document.childNodes[0];
            // Lấy node của của <html>
            var docnodes = entiredoc.childNodes;
            // Duyệt và alert Tên node con của <html>
            for(i = 0; i < docnodes.length; i++)
                alert(docnodes[i].nodeName);
        }

        /* Hàm cần viết: duyệt toàn bộ node trong cây DOM */
        // Input: 1 node parent
        // Output: in ra tất cả node con
        // Hàm này thực hiện đệ quy
        function duyetDom(parent) {
            // Nội dung hàm cần xử lý
            // Điều kiện dừng
            // In ra tên node
            // Lấy node con của node parent
            // Duyệt từng node con
        }
    </script>
</head>
<body name="bodybody">
    <script> appendp(); </script>
    <!-- Gán gọi hàm vào button -->
    <input type="button" value="Display children of HTML tag" onclick="duyetDom(document.childNodes[0]);"/>
</body>
```

Khi mở file SOP_DOM_Task_1.html trên trình duyệt (firefox):

Kết quả khi nhấn vào nút: "Display children of HTML tag"

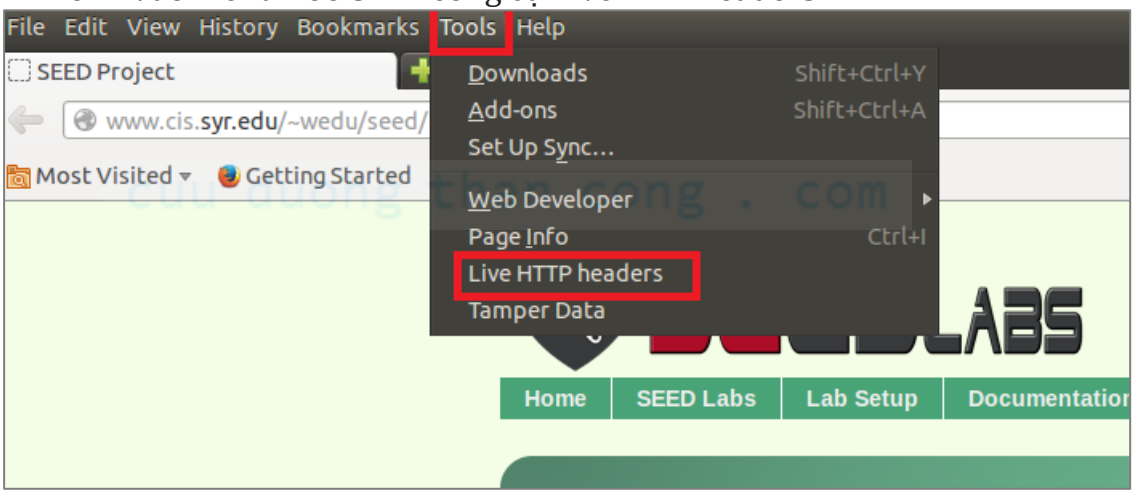


b. Ứng dụng web Collabtive sử dụng cơ chế quản lý phiên (session) dựa trên cookie. Xác định tên cookie trong Collabtive (www.soplabcollabtive.com) sử dụng Live HTTPHeaders extension (chụp ảnh màn hình).

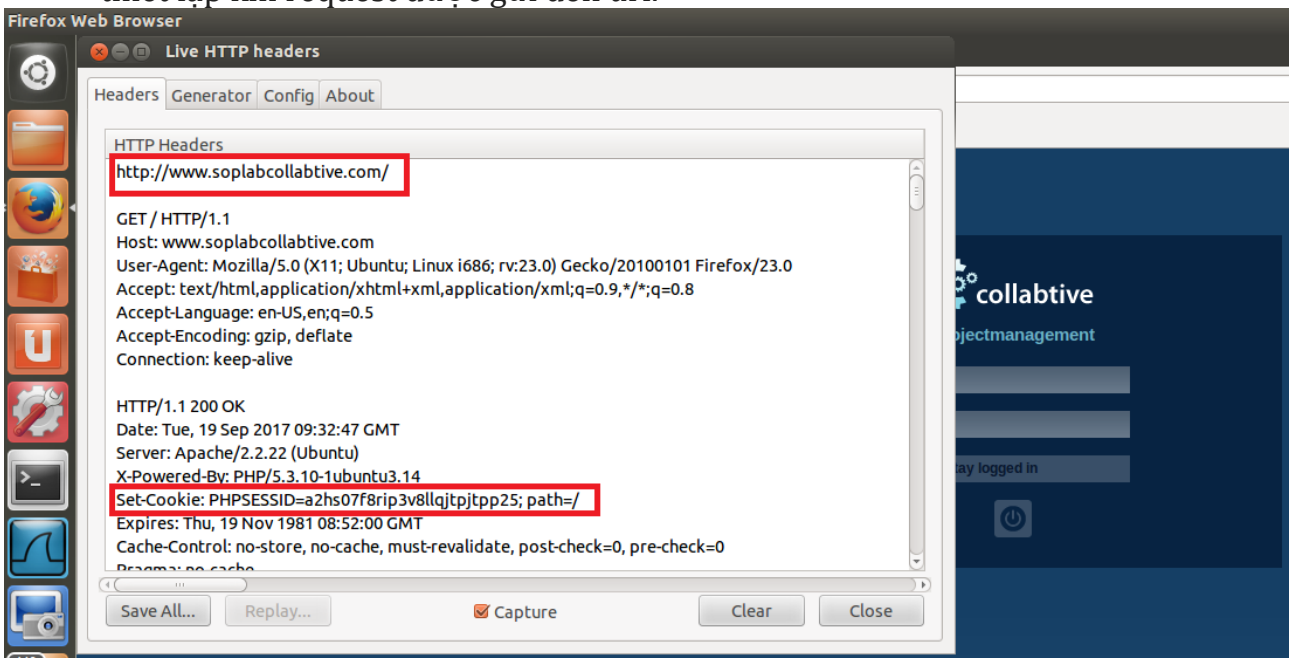
Hướng dẫn:

Bước 1: Mở trình duyệt firefox.

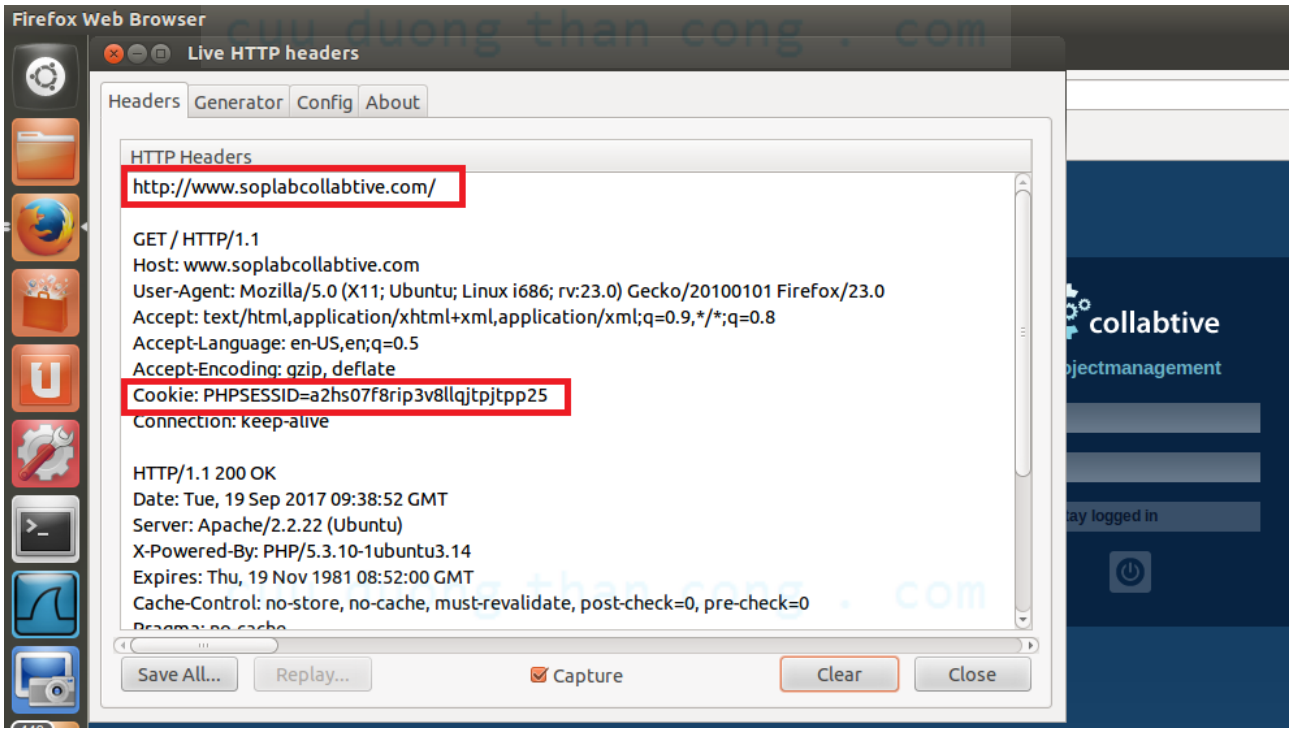
Bước 2: Vào Menu Tools mở công cụ LiveHTTPheaders.



Bước 3: truy cập vào url: www.soplabcollabtive.com. Cookie PHPSESSID được thiết lập khi request được gửi đến url.



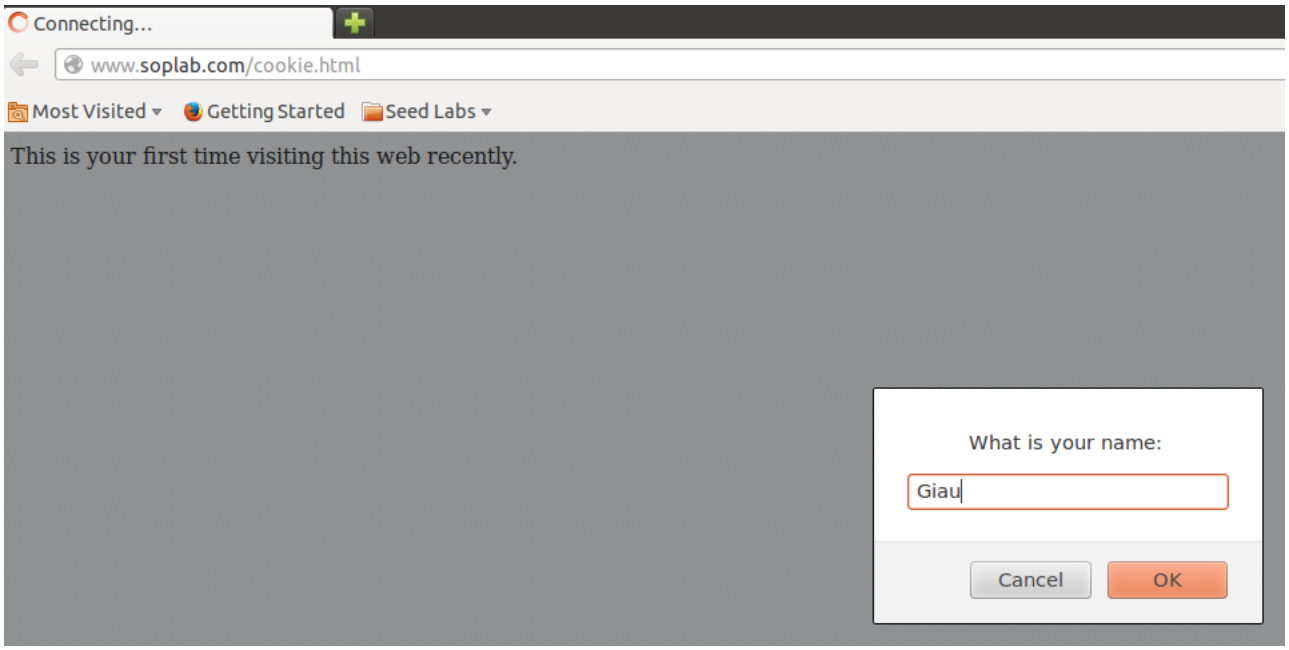
Bước 4: Hình ảnh cookie được thiết lập cho các lần kế tiếp khi truy cập vào url này.



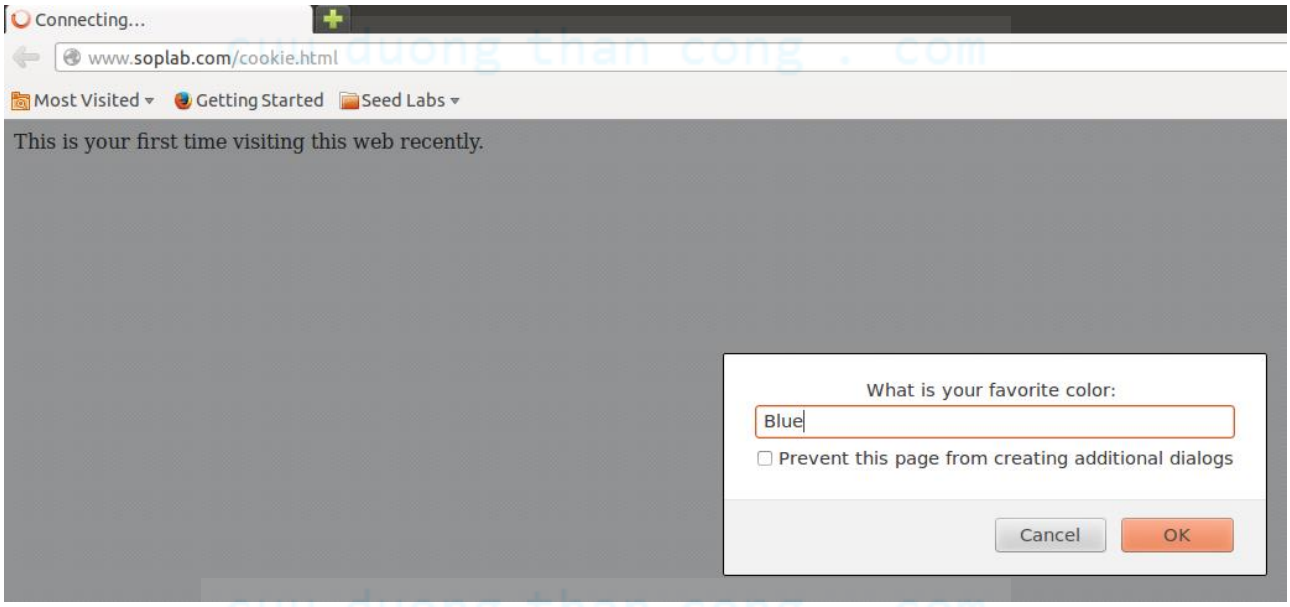
c. Đọc mã nguồn www.soplab.com/cookie.html để hiểu cách lưu trữ, đọc và xử lý cookie. Viết một đoạn JavaScript trong file cookie.html để hiển thị số lần đã truy cập của người dùng hiện tại.

Hướng dẫn:

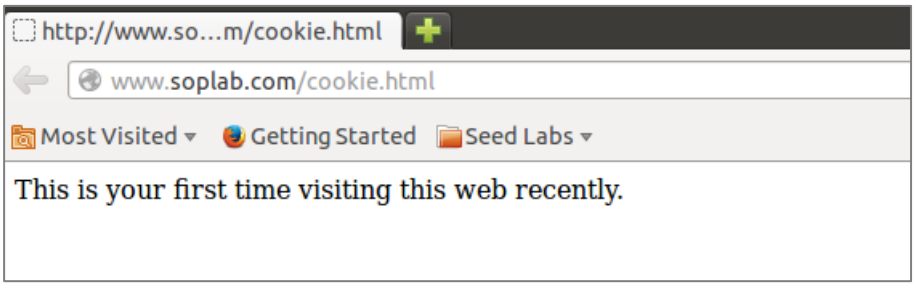
Bước 1: Mở trình duyệt Firefox và truy cập vào url www.soplab.com/cookie.html. Bạn được yêu cầu nhập tên và màu sắc yêu thích.



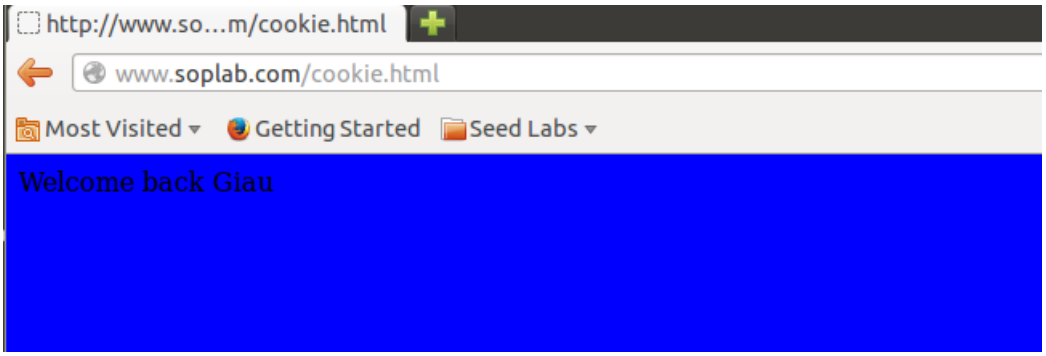
Kế đến là nhập màu sắc yêu thích.



Và màn hình lần đầu tiên bạn ghé thăm trang web này.



Bước 2: Truy cập hoặc refresh lại trang web. Bạn sẽ thấy một dòng thông báo chào bạn đã quay lại cùng màu nền là màu bạn yêu thích (nếu đánh đúng và bằng tiếng Anh).

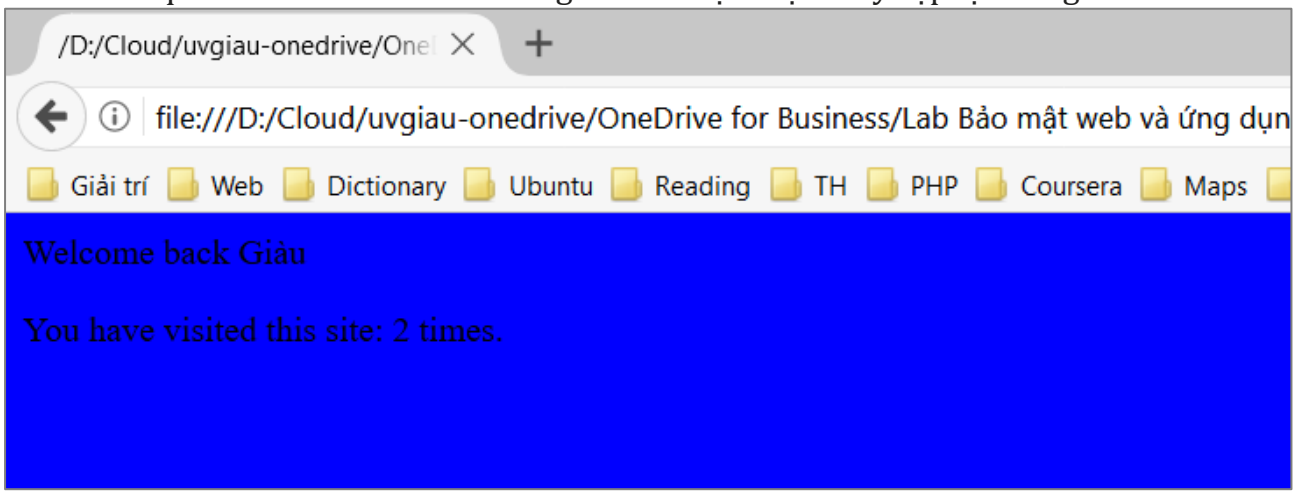


Bước 3: Vào đường dẫn /var/www/SOP và mở file cookie.html bằng editor để xem mã nguồn. Sau đó thực hiện chỉnh sửa mã nguồn để các lần kế tiếp truy cập vào url này thì sẽ hiện số lần truy cập hiện tại.
Hướng dẫn chỉnh sửa mã nguồn:

*cookie.html

```
<script>
var simplecookie = document.cookie;     // get the cookie

var age = (60*60);      // the cookie will expire in one hour
var path = "/cookie";           // the cookie is shared with all directories in the same domain
if (simplecookie == "") // store the cookie if it's not already set
{
        document.write("This is your first time visiting this web recently.");
        var name = encodeURIComponent(prompt("What is your name: "));   // value shouldn't contain ";" or other special
        var color = encodeURIComponent(prompt("What is your favorite color: "));
        document.cookie = "name=" + name +      // '+' concatenates strings together.
                "&color=" + color +             // different properties of cookie are seperated by "&".
        // Cần lưu thêm một biến điếm số lần truy cập: visitCounter
                ";max-age=" + age;              // different attributes are seperated by ";".
        //      ";path=" + path;                Path attribute doesn't work as expected. Try it out yourself.
        // we didn't set domain attribute since we only share the cookie within the current domain
        // we didn't set secure attribute since it's ok to transmit the cookie in plain text.
}
else
{
        var cookies = simplecookie.split('&');      // Break the string of all cookies into individual cookie strings
        for(var i = 0; i < cookies.length; i++)         // parse cookies.
        {
                if (cookies[i].substring(0, "name=".length) == ("name="))
                {
                        var name = decodeURIComponent(cookies[i].substring("name=".length));
                        document.writeln("Welcome back " + name + "<p>");
                }
                else if (cookies[i].substring(0, "color=".length) == ("color="))
                {
                        var color = decodeURIComponent(cookies[i].substring("color=".length));
                        document.bgColor = color;       // set background color as user's preference.
                }
                // Cần xử lý tăng số lần truy cập cho biến visitCounter
        }

        document.cookie = simplecookie;         // reset cookie to refresh age.
        // Xử lý thay thế và lưu lại cookie có biến đếm số lần truy cập
}
</script>
```

Kết quả sau khi chỉnh sửa mã nguồn và thực hiện truy cập lại trang web:



2. SOP cho DOM và Cookies

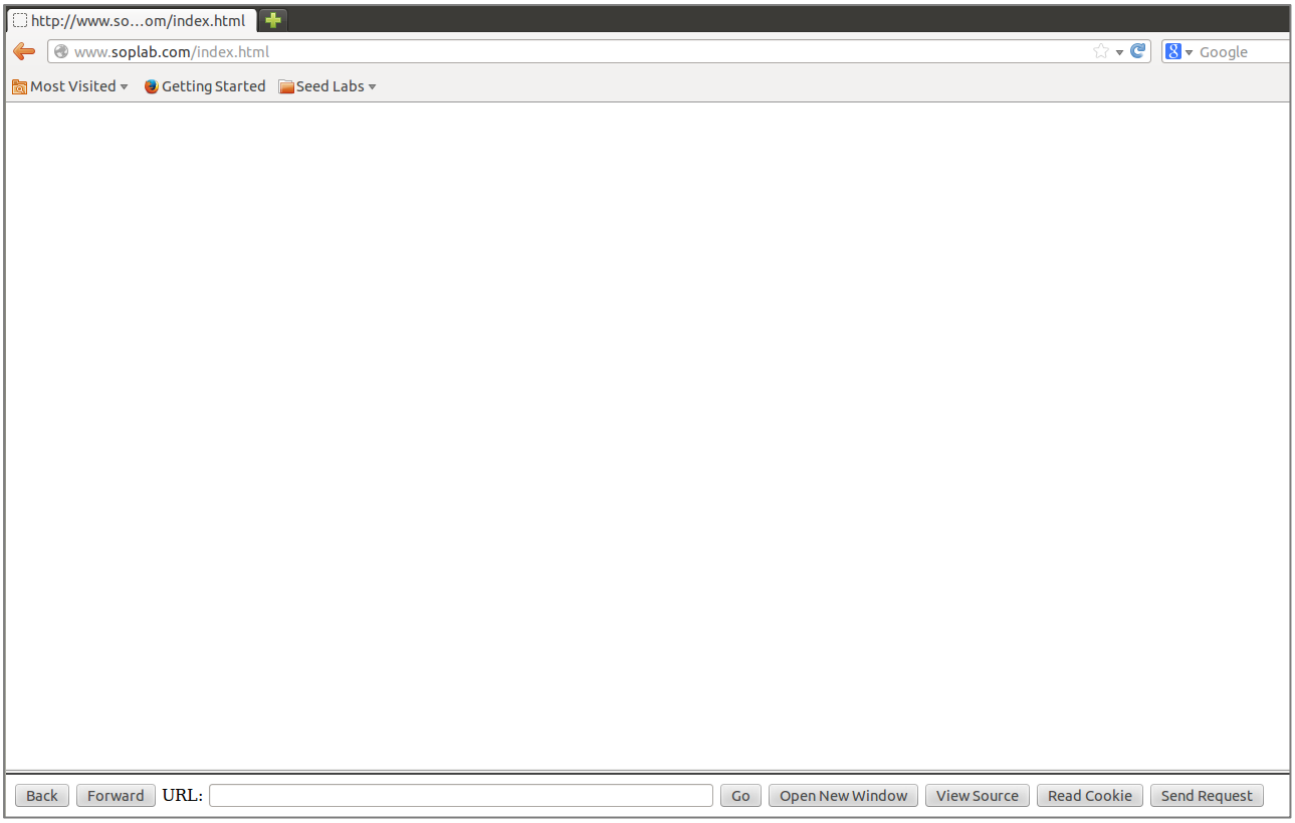
Mục tiêu: minh họa cách trình duyệt nhận biết nguồn gốc của ứng dụng web và cách hạn chế truy cập được áp dụng trên DOM và Cookies.

Để minh họa SOP cho DOM và Cookies, sử dụng trang web www.soplab.com/index.html. Trang web hiển thị 2 trang web bên trong frame.

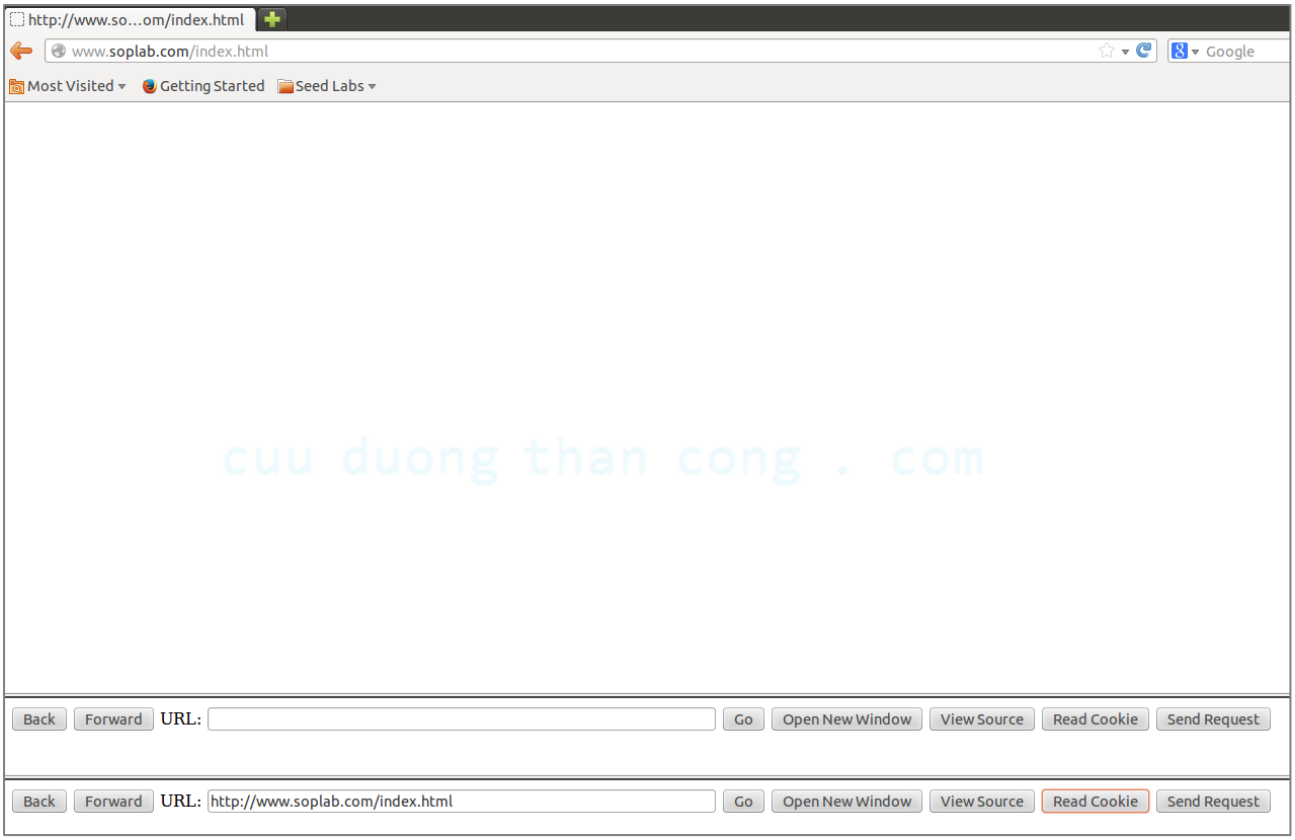
```
<frameset rows=”*, 75”>
        <frame src=”about:blank” name=”main”>
        <frame src=”navigation.html”>
</frameset>
```

Frame đầu tiên hiện thị nội dung trang web www.soplab.com/navigation.html. Trang web này yêu cầu cung cấp một URL khác để hiện thị cho frame còn lại. Trong trang web navigation.html có 2 đoạn mã JavaScript dùng để xem Mã nguồn (View Source) và đọc Cookie (Read Cookie) của URL được cung cấp. Có nghĩa là mã JavaScript của trang navigation.html có thể đọc được DOM và Cookie của frame còn lại. Điều này, cho thấy rằng, chúng ta có một trang web có thể truy cập vào tài nguyên của một trang web khác. Thực hiện bài lab để hiểu được hạn chế truy cập vào DOM và Cookie dựa trên SOP.

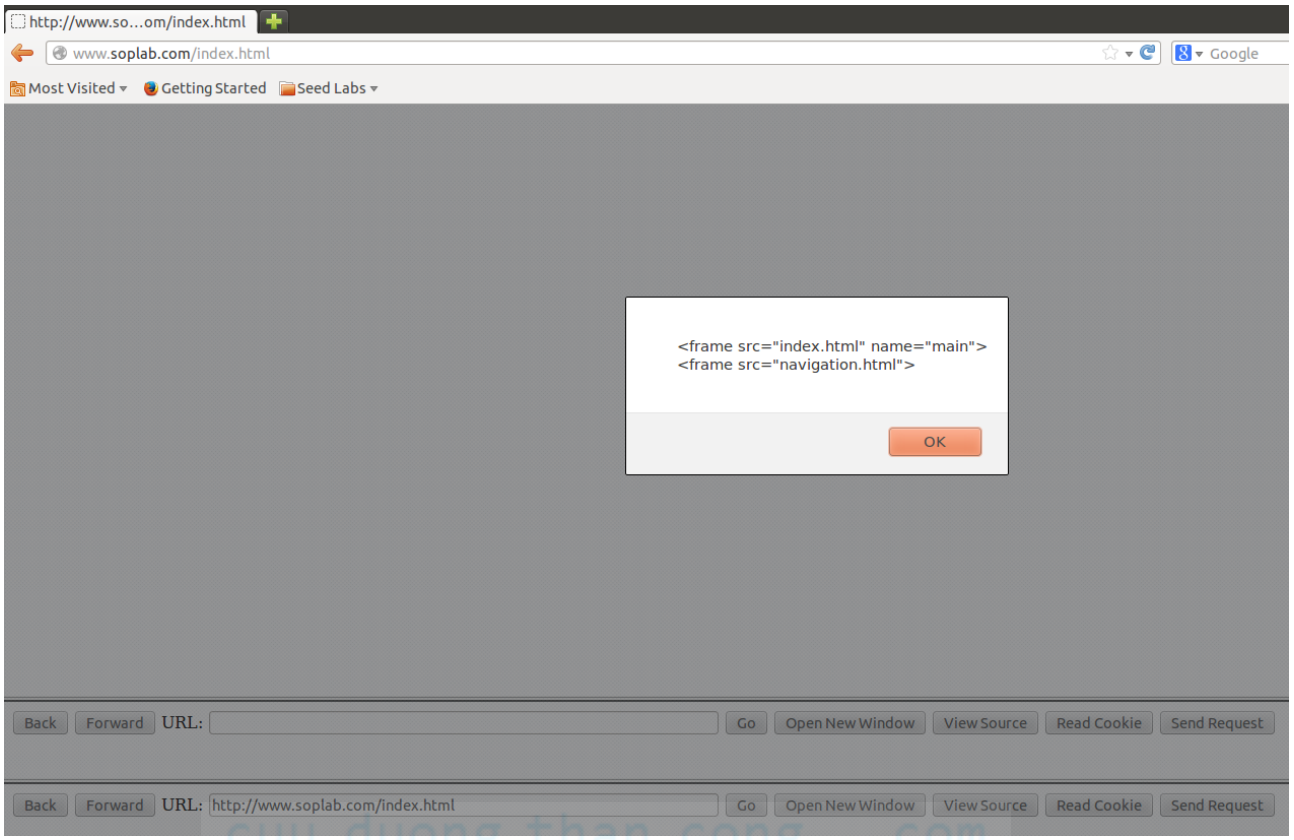
a. Truy cập trang web www.soplab.com/index.html và lần lượt cung cấp các URL sau vào input URL và xem bạn có thể truy cập vào cookies và DOM của các trang web đó không?
   http://www.soplab.com/index.html
   http://www.soplab.com/navigation.html
   Thử sử dụng một vài website khác như http://tuoitre.vn/ và xem bạn có thể đọc cookies và DOM không?
   Chụp lại màn hình.
   Hướng dẫn:
   Bước 1: Mở trình duyệt vào truy cập vào url www.soplab.com/index.html.



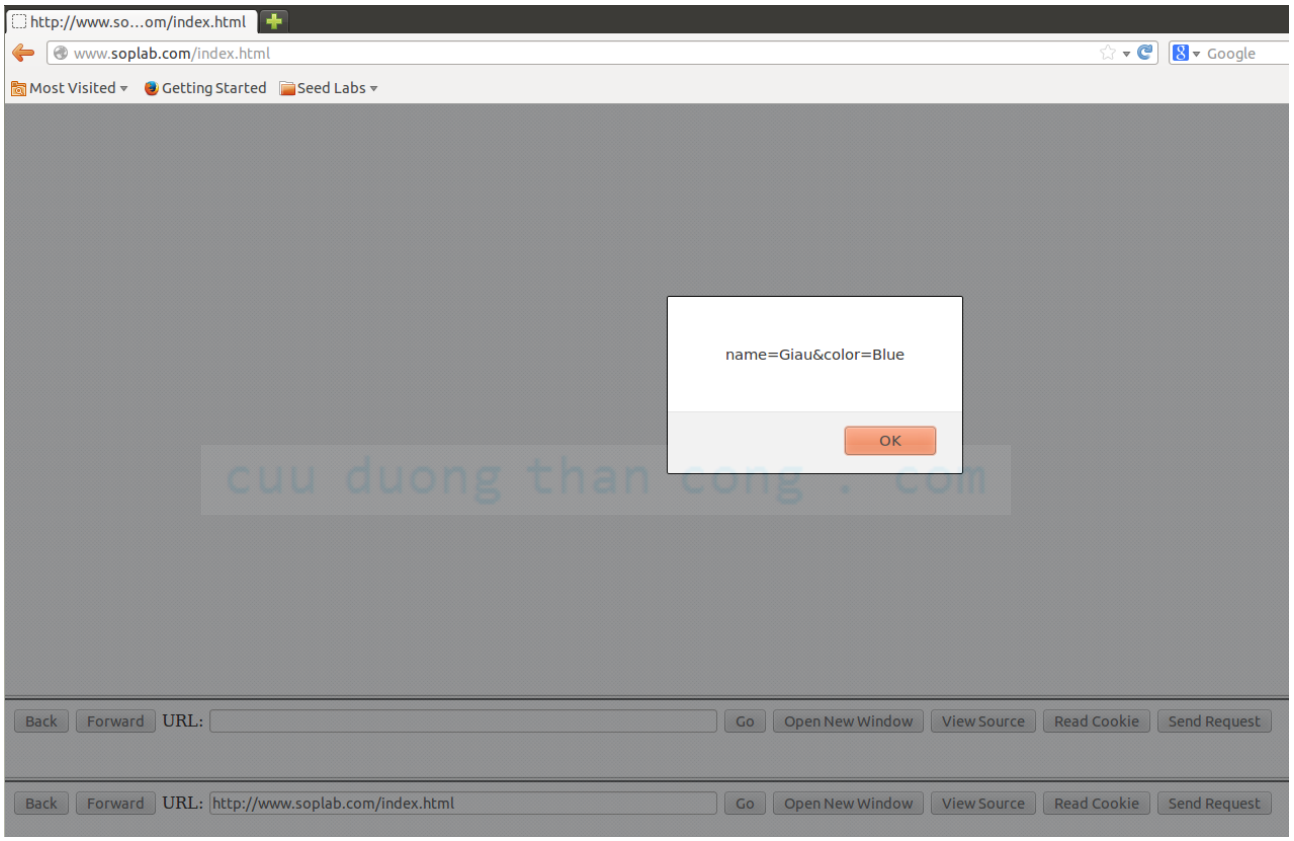
Bước 2: Lần lượt nhập url đã cho vào khung URL và nhấn nút Go. Tại mỗi bước thực hiện bấm View Source và Read Cookie (Lưu ý: nút của frame dưới cuối màn hình). Ghi nhận lại kết quả.
Kết quả khi cung cấp URL www.soplab.com/index.html.



Khi truy cập vào www.soplab.com/index.html và nhấn vào View Source.



Khi truy cập và www.soplab.com/index.html và nhấn vào Read Cookie.



Bước 3: Thực hiện lại bước 2 cho các URL còn lại.

b. Web server đang sử dụng 2 port là 80 và 8080. Nhập http://www.soplab.com:8080/navigation.html vào URL và xem bạn có thể đọc cookies và DOM.
Hướng dẫn:
Thực hiện tương tự bước 2 ở phần a.

c. Không chỉ cookies và nội dung của frame bị hạn chế của SOP, mội vài đối tượng khác cũng bị hạn chế như History, URL của frame. Truy cập vào trang www.soplab.com/index.html. Lần lượt nhập các link dưới vào URL, nhấn Go rồi nhấn Back. Chụp màn hình và nhận xét:

www.soplab.com/navigation.html
http://tuoitre.vn/
http://www.soplab.com:8080/navigation.html

Hướng dẫn:
Bước 1: Truy cập vào trang www.soplab.com/index.html
Bước 2: Nhập từng link vào khung URL, nhấn Go để truy cập. Sau đó, bấm nút Back và xem trình duyệt có gì thay đổi không?

3. SOP cho XMLHttpRequest

Mục tiêu: Hiểu được SOP cho XMLHttpRequest.

Xem ví dụ cách sử dụng HMLHttpRequest tại hàm sendRequest() của file /var/www/SOP/navigation.html.

Thực hiện yêu cầu sau:

a. Truy cập vào trang www.soplab.com/navigation.html. Lần lượt nhập các link sau vào khung URL và nhấn Send Request. Ghi nhận kết quả và nhận xét.

www.soplab.com/navigation.html
http://tuoitre.vn/
http://www.soplab.com:8080/navigation.html

Hướng dẫn:
Truy cập vào trang www.soplab.com/navigation.html. Sau đó, lần lượt nhập các url rồi nhấn Send Request. Chụp màn hình và nhận xét (có thể nhận xét tổng thể).

b. Giả sử HTTP request sử dụng XMLHttpReques API không được áp dụng SOP (Có nghĩa là bạn có thể truy cập vào tài nguyên của trang khác). Bạn hãy mô tả một vài tấn công có thể xảy ra.
Hướng dẫn: tìm một ví dụ hoặc cách tấn công cụ thể khi không có sự hạn chế truy cập tài nguyên từ một trang web khác.

4. Ngoại lệ trong SOP

Một vài thẻ HTML có thể gửi yêu cầu HTTP request từ trong trang web đến trang web khác. Ví dụ, thẻ img.

Hãy kiểm tra những thẻ sau (frame, iframe, img, a), quan sát và báo cáo.

Hướng dẫn:

Bước 1: Viết một đoạn code HTML ngắn gọn có sử dụng thẻ yêu cầu.

```
<!DOCTYPE html>
<html>
    <head>
        <title></title>
    </head>
    <body>
        <img src="http://chohoaviet.com/wp-content/uploads/2016/04/hoa-tigon11.jpg">
    </body>
</html>
```

Bước 2: Mở đoạn code vừa viết bằng trình duyệt, chụp màn hình.

Bước 3: Giải thích thẻ và kết luận.

C. YÊU CẦU

- Sinh viên tìm hiểu và thực hành theo hướng dẫn.
- Nộp báo cáo kết quả gồm chi tiết những việc bạn đã thực hiện, quan sát thấy và kèm ảnh chụp màn hình kết quả (nếu có); giải thích cho quan sát (nếu có).
- Sinh viên báo cáo kết quả thực hiện và nộp bài.

Báo cáo:

- File .PDF. Tập trung vào nội dung, không mô tả lý thuyết.
- Đặt tên theo định dạng: [Mã lớp]-LabX_MSSV1-Tên SV.

Ví dụ: [NT101.H11.1]-Lab1_14520000-NguyenVanA.

- Nếu báo cáo có nhiều file, nén tất cả file vào file .ZIP với cùng tên file báo cáo.
- Nộp file báo cáo trên theo thời gian đã thống nhất tại courses.uit.edu.vn.

Đánh giá: Sinh viên hiểu và tự thực hiện được bài thực hành. Khuyến khích:

- Chuẩn bị tốt và đóng góp tích cực tại lớp.
- Có nội dung mở rộng, ứng dụng trong kịch bản phức tạp hơn, có đóng góp xây dựng bài thực hành.

Bài sao chép, trễ, … sẽ được xử lý tùy mức độ vi phạm.

D. THAM KHẢO

[1] SEED Lab, http://www.cis.syr.edu/~wedu/seed/web_security.html
[2] DOM, https://www.w3schools.com/js/js_htmldom.asp
[3] DOM, https://www.w3schools.com/jsref/dom_obj_document.asp
[4] SOP, https://www.w3.org/Security/wiki/Same_Origin_Policy

HẾT