<https://viblo.asia/p/su-dung-react-de-goi-den-cross-domain-server-YrEBRA7VG8Zj>

**Intro**

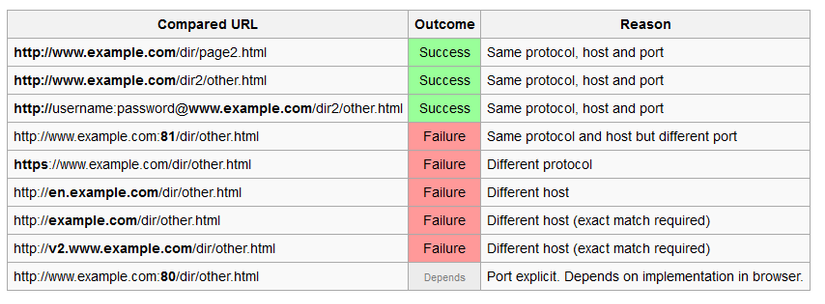
Cùng tìm hiểu về Same Origin Policy, CORS và cách gọi từ ReactJS qua bài blog sau nhé 

**Same Origin Policy**

Trong khoa học máy tính, **same-origin policy** là một khái niệm quan trọng trong mô hình bảo mật thông tin của web application. Dưới chính sách này, các trình duyệt web chỉ cho phép các đoạn script ở trang web thứ nhất truy cập vào dữ liệu ở trang web thứ hai (AJAX request), chỉ khi mà hai trang đó có cùng nguồn (same-origin). Việc cùng nguồn gốc hay không được định nghĩa thông qua một nhóm các yếu tố: đó là **URI**, **hostname**, **port number**. Lợi ích của việc này chính là ngăn cản các script độc hại ở một trang có thể có được các thông tin nhạy cảm ở một trang khác thông qua việc xử lý trên DOM (Document Object Model) (from Wikipedia). Tức là:

* Tức là ta có thể nhúng ảnh, video, css, javascript từ các trang khác vào
* Nhưng bị hạn chế khi thực hiện AJAX lên các domain khác (tưởng tượng sẽ nguy hiểm như thế nào nếu một đoạn script độc hại lấy thông tin của trang rồi POST thông tin đó lên một domain khác do hacker quản lý)

Chúng ta cùng xem qua hình sau để thấy rõ được trong những trường hợp URI nào thì ta có thể gọi được khi ta truy cập từ trang <http://www.example.com/dir/page.html>



Cần chú ý rằng **Same Origin Policy** là do các trình duyệt hỗ trợ chứ không phải là do phần nào khác. Các trình duyệt phổ biến hiện nay như: Firefox, Chrome, Safari,... đều có hỗ trợ tính năng này.

Tuy rằng nâng cao bảo mật nhưng chính sách này vẫn có một số hạn chế. Chẳng hạn sẽ có những nguồn tài nguyên mà mục đích được đưa lên trên Internet là để cho tất cả mọi người cùng truy cập một cách công khai, có thể kể đến là Google Web Font.

Trong bài này ta sẽ nói về một trong những phương pháp để làm lỏng chính sách này, đó là : **Cross Origin Resource Sharing**

**Cross Origin Resource Sharing (CORS)**

Để thực hiện được CORS trên các trình duyệt (CORS được hỗ trợ bởi hầu hết các trình duyệt phổ biến hiện nay: bắt đầu từ Firefox 3.5, Safari 4, và Chrome 3. Internet Explorer 10 hiện tại cũng đã hỗ trợ hoàn toàn) thì ta cần sử dụng thêm các header **Origin** request header và **Access-Control-Allow-Origin** header. Cùng xem qua ví dụ đơn giản sau để hiểu thêm.

* Chẳng hạn, một đoạn script từ trang [http://example.com](http://example.com/) dùng AJAX để truy cập vào tài nguyên ở trang <http://otherdomain.com/some-resource>.
* Nếu là code JQuery ta sẽ viết như sau

$.ajax({

url: "http://otherdomain.com/some-resource",

type: "GET",

success: function(data) {

},

error: function(e) {

console.log(e);

}

});

* Lúc đó trình duyệt sẽ tự động sinh thêm header **Origin** có giá trị là domain của chúng ta. Header này chỉ được sinh ra khi đích đến của request là cross-origin.

GET http://otherdomain.com/some-resource/ HTTP/1.1

Referer: http://example.com/my-app/

Origin: http://example.com

* Nếu là một server chấp nhận CORS thì server sẽ trả về response như sau:

Access-Control-Allow-Origin: http://example.com

Content-Type: application/json

* **Access-Control-Allow-Origin** header chỉ định domain được phép truy cập. Trình duyệt kiểm tra, nếu trùng với domain hiện tại ([http://example.com](http://example.com/)) sẽ cho phép xử lý response trả về.

Đối với các request phức tạp hơn, hãy tham khảo ở <http://techblog.constantcontact.com/software-development/using-cors-for-cross-domain-ajax-requests/> và <http://www.html5rocks.com/en/tutorials/cors/>