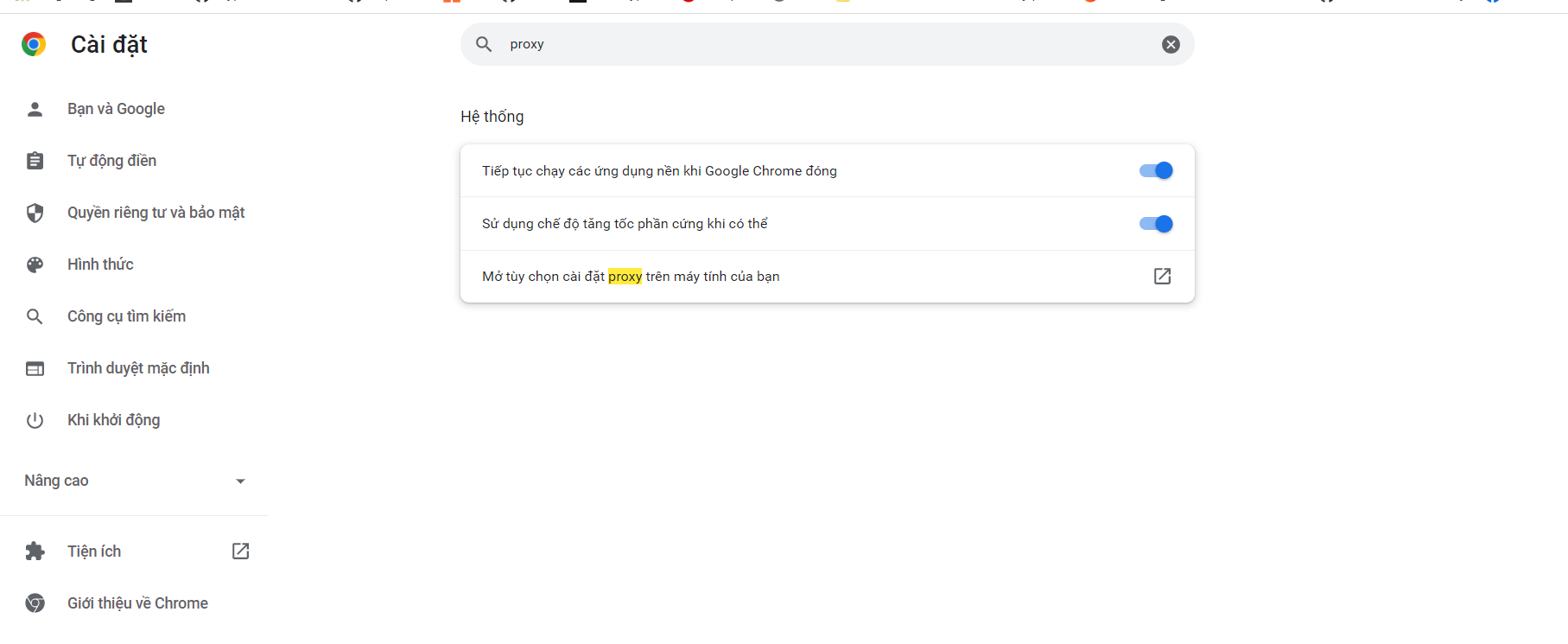
Web Application Technologies

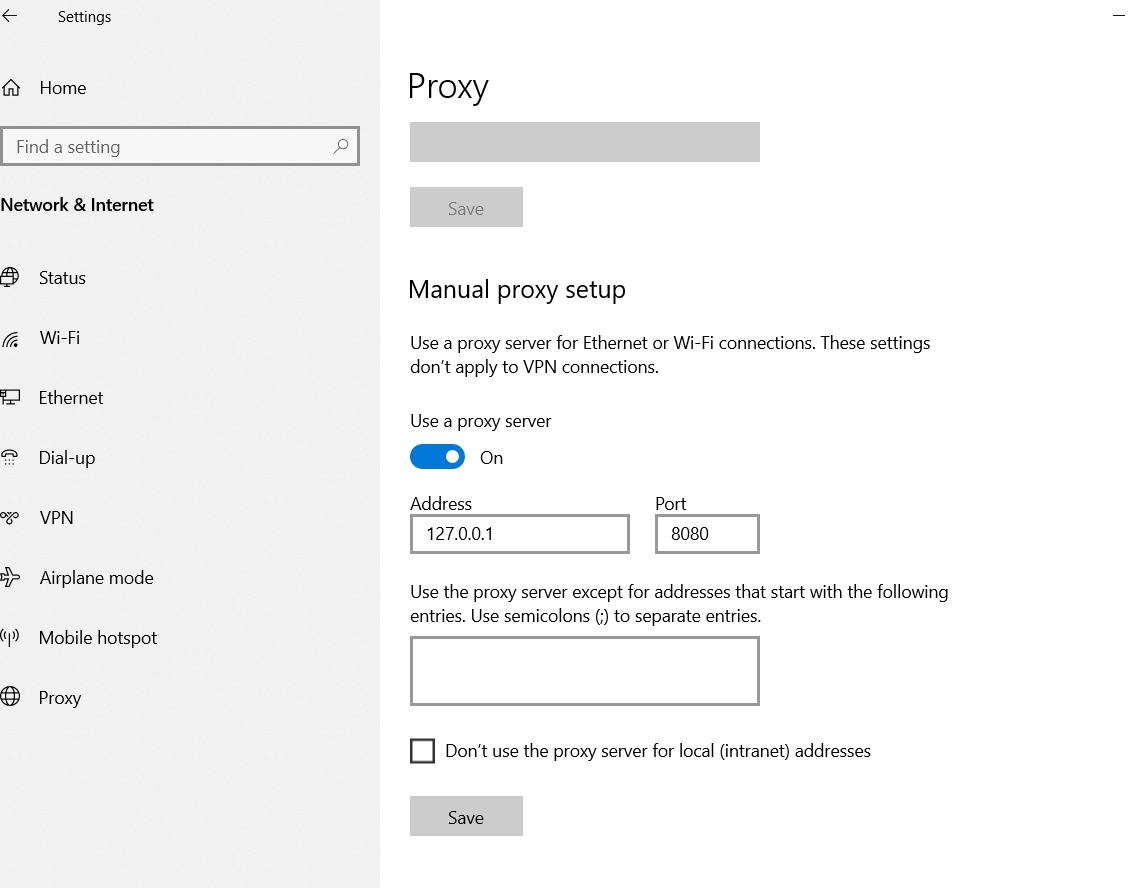
1. Cài đặt và làm quen với công cụ burpsuite

- Cài đặt proxy cho trình duyệt

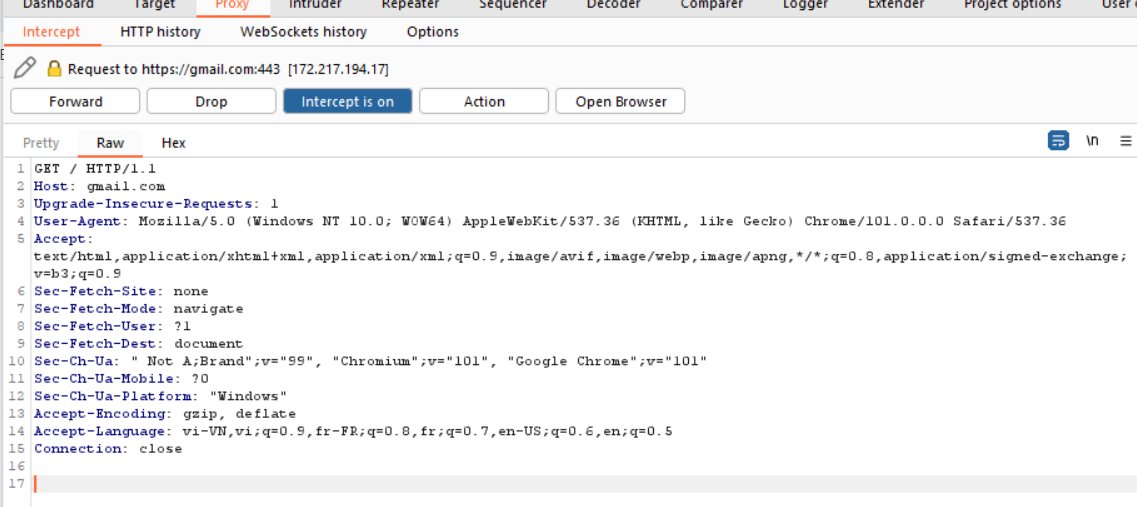
B1: Thực hiện trên trình duyệt chrome đầu tiên ta vào Cài đặt->Trên thanh tìm kiếm nhập proxy và chọn vào phần thiết lập proxy



B2: Sau đó ta sẽ click vào Use a proxy server để enable và điền vào ip và port của proxy và save.

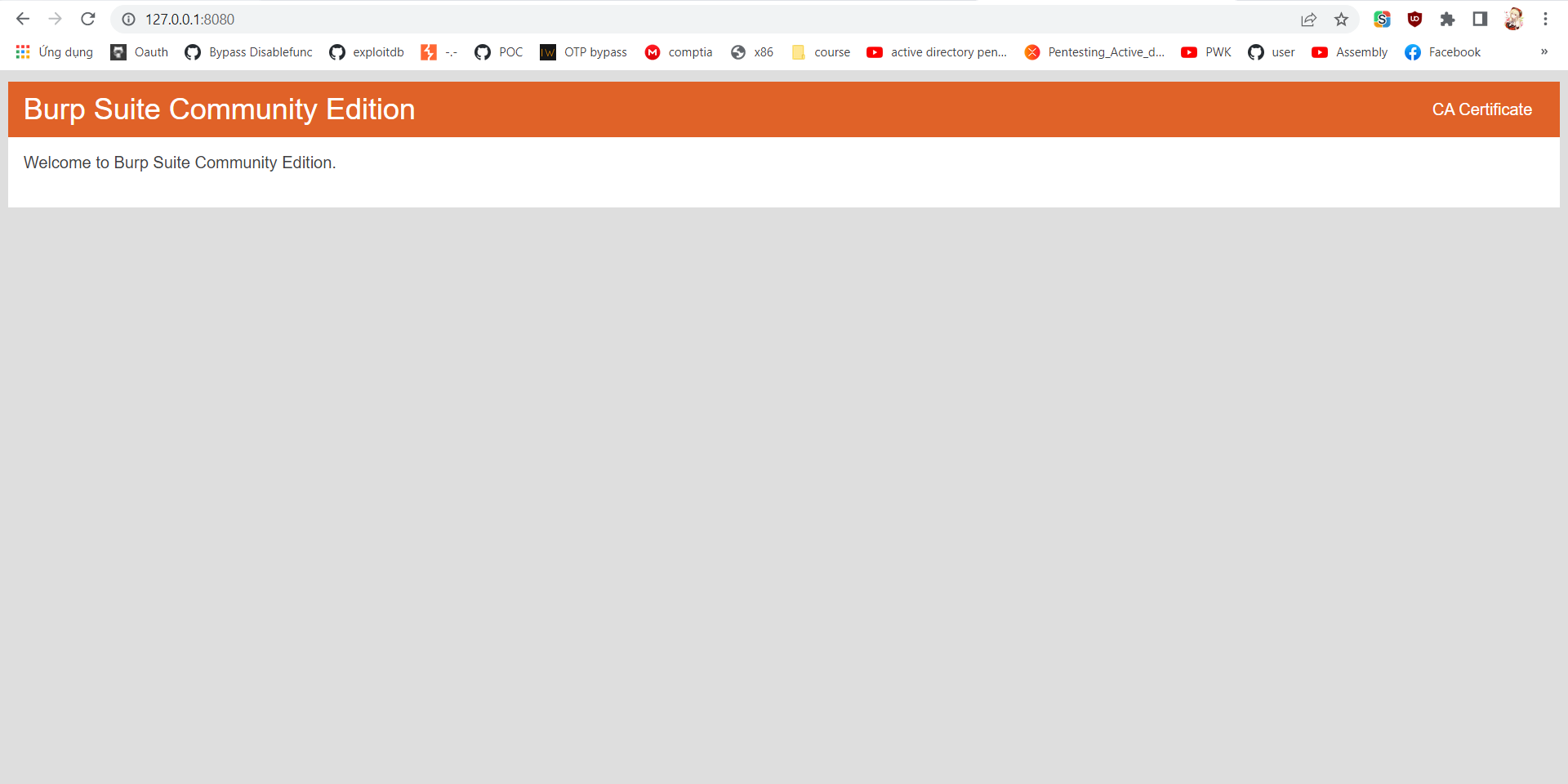


Như vậy ta đã cấu hình thành công ta sẽ kiểm tra bằng cách dùng công cụ burpsuite và bắt request. Sau đây là một request gửi đến trang gmail.com đi qua proxy



-Cài đặt certificate cho các trình duyệt khác nhau

B1: đầu tiên ta sẽ tải certificate của burpsuite qua địa chỉ <http://127.0.0.1:8080/> và click vào CA certificate



B2: sau đó click chuột phải vào file CA đó và chọn install certificate nó sẽ xuất hiện chương trình cài đặt và ta chọn open sau đó ta chọn local machine click next và chọn Place all certificate in the following store và browse vào Trusted Root Certification Authorities sau đó click OK \_> Next và finish. Như vậy ta đã thành công cài đặt certificate.

-Các chức năng của Burpsuite:

Decoder: là chức năng giúp giải mã (decode) và mã hóa (encode) các chuỗi theo nhiều dạng format khác nhau. Để sử dụng Decoder, bạn vào tab decoder và nhập các dữ liệu cần giải mã hoặc mã hoá.

Proxy Server được thiết kế để bắt các yêu cầu và tuỳ ý sửa đổi chúng trước khi gửi lên máy chủ. Người dùng cần bật Burp Suite và vào tab proxy, sau đó option đánh dấu vào ô Running để sử dụng chức năng này.

Repeater: Chức năng Repeater cho phép người dùng sử dụng request được gửi đến máy chủ và có thể nhanh chóng sửa đổi nội dung yêu cầu một cách tuỳ ý trong nhiều lần khác nhau.

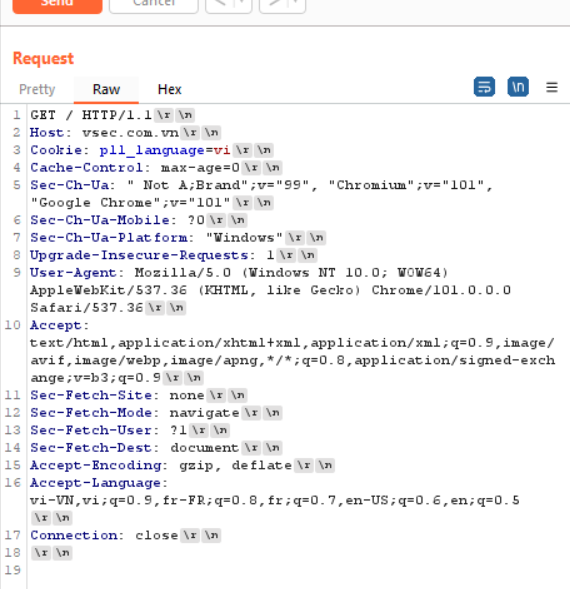
Comparer: Comparer là tính năng được sử dụng để phân biệt được sự khác nhau giữa các yêu cầu (requests) và phản hồi (reponses), so sánh một cách dễ dàng các tệp tin và các dữ liệu có dung lượng lớn.

Web spider: Web spider là tính năng duyệt web tự động được dùng với mục đích xác định cấu trúc của một trang web. Để có thể sử dụng tính năng này, bạn cần vào tab Sitemap và Target, sau đó thực hiện thao tác click chuột phải vào mục tiêu và lựa chọn spider cho máy chủ.

1. Thành phần của một HTTP Request/HTTP Response

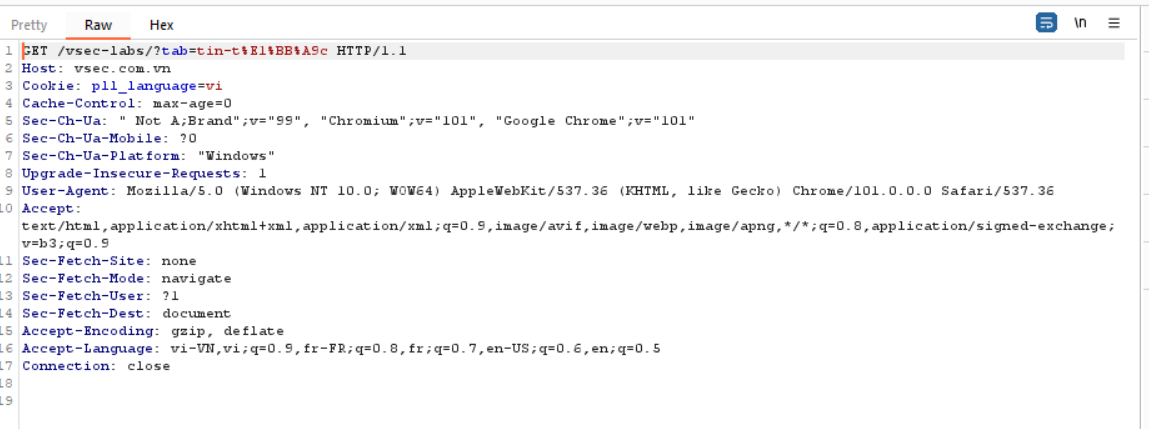
-Các blank line trong một request

Là các ký tự \r\n thường được sử dụng để kết thúc mỗi dòng header trong request và \r\n\r\n để kết thúc một request



-Tìm hiểu và demo các http method trên website:

Get request: được sử dụng để lấy thông tin từ server theo URL đã cung cấp như ở trên ta lấy tài nguyên thông qua param tab



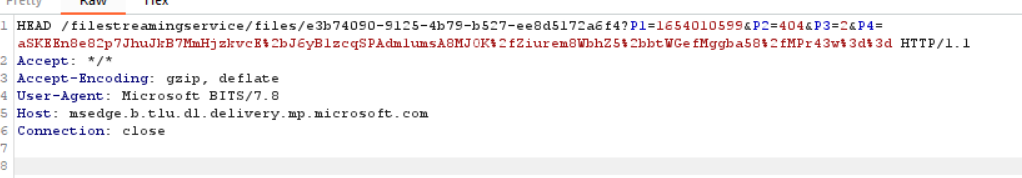
Post request: Sử dụng để gửi thông tin user qua các biểu mẫu. Như hình dưới ta sẽ gửi data ở trong body request với user và password.



PATCH: ghi đè các thông tin được thay đổi của đối tượng. ở đây là đang ghi đè avatar khi người dùng thay đổi avatar của họ.



HEAD: giống với GET nhưng response trả về không có body, chỉ có header.



PUT: ghi đè tất cả thông tin của đối tượng với những gì được gửi lên.

CONNECT: thiết lập một kết nối tới server theo URI.

OPTIONS: mô tả các tùy chọn giao tiếp cho resource.

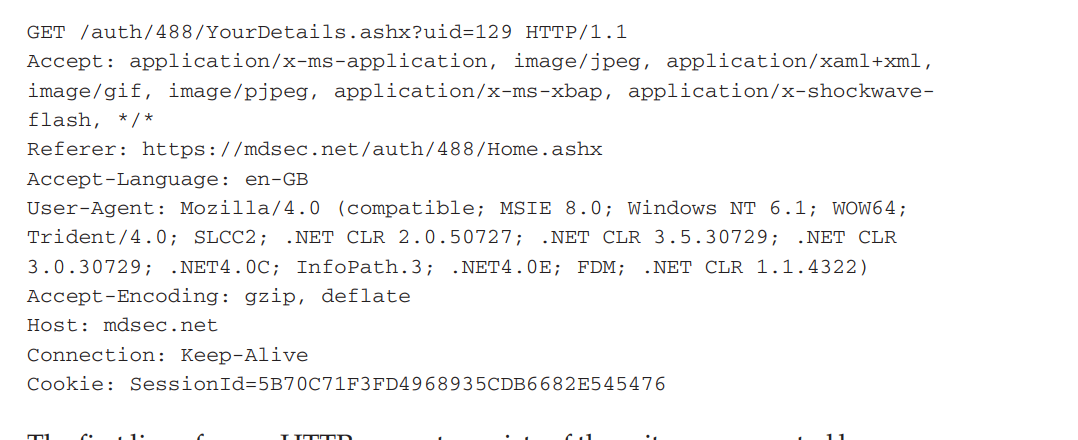
TRACE: thực hiện một bài test loop - back theo đường dẫn đến resource.

- Các HTTP Header - Tìm hiểu kỹ Host Header

**HTTP header**

Tiêu đề http cho phép máy khách và máy chủ chuyển thông tin với nhau với một http request hay http respond. Tất cả thông báo http sẽ gồm một hoặc nhiều tiêu đề trên từng dòng riêng biệt kết thúc bằng khoảng trắng(\r\n) và kết thúc một request sẽ bằng ký tự(\r\n\r\n).

VD: HTTP header



Phần đầu tiên của HTTP request luôn có 3 phần phân biệt bằng khoảng cách

* Phần đầu tiên là http method sẽ chỉ ra mà method của request đang sử dụng như trên vd là GET
* Phần tiếp theo sẽ là url được yêu cầu là nơi muốn lấy tài nguyên đi kém với nó sẽ là các param được người dùng yêu cầu các param sẽ được xác định bằng dấu "?".
* Tiếp theo sẽ là phiên bản http được sử dụng. Có 2 phiên bản được sử dụng phổ biến là version 1.0 và 1.1. Sự khác biệt cần lưu ý giữa 2 phiên bản là version 1.1 host header reuqest là bắt buộc.

+Referer header: Địa chỉ của trang web trước đó mà từ đó có một liên kết đến trang được yêu cầu hiện tại.

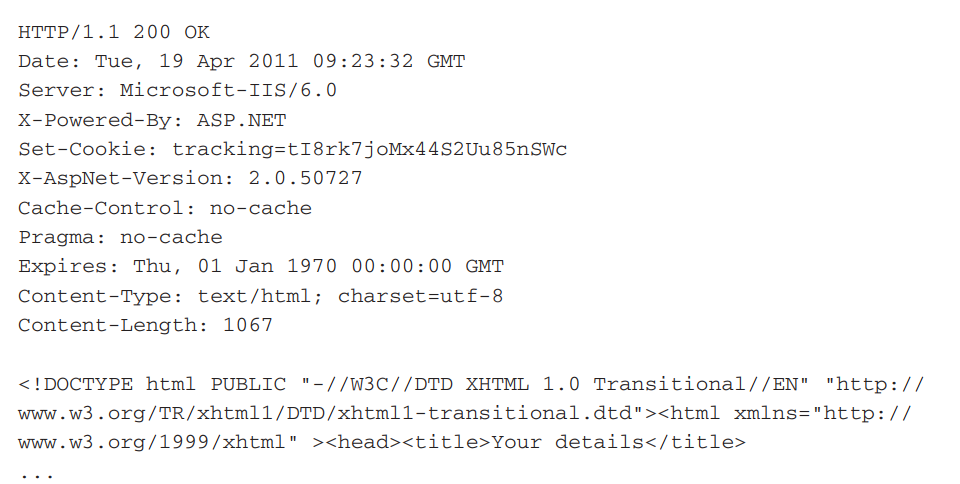
VD: nếu bạn click vào link trên site.com/page để truy cập another.com/page giá trị referer nhận được bởi trang another.com/page sẽ có giá trị là site.com/page.

+User-Agent header được sử dụng để cung cấp thông tin về trình duyệt hoặc phần mềm tạo ra request. note(<https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/User-Agent/Firefox>).

+Host header: chỉ định tên miền của máy chủ.

+Cookie header: được sử dụng để server gửi các tham số bổ sung máy chủ đã phát hành cho client.

**HTTP Responses**



Dòng đầu tiên của http response gồm ba mục phân biệt bằng dấu cách.:

* http version đang sử dụng.
* status code. Tùy vào từng status code mà sẽ có những mã khác nhau.
* reason phrase dùng để mô tả về status code.

Server header: được dùng để mô tả phần mềm được sử dụng bởi máy chủ đã tạo response.

Set-cookie header: được máy chủ gửi cho người dùng , để người dùng lưu lại và dùng nó để gửi lên cho máy chủ sau này.

Pragma header: sẽ hướng dẫn trình duyệt không lưu response trong bộ nhớ đệm.

Expires: Tiêu đề HTTP Expires chứa ngày / giờ mà sau đó phản hồi được coi là hết hạn.

Content-Type header: sẽ chỉ ra media của tài nguyên lúc ban đầu.

Content-Length: chỉ ra độ dài nội dung bao nhiêu byte.

HTTP hỗ trợ một số lượng lớn header, mỗi header được thiết kế cho các mục đích khác nhau. Một số được áp dụng cho cả request và response.

Sau đây là một số header thường gặp khi tấn công:

Connection header: sẽ cho biết là kết nối này sẽ đóng hay vẫn tiếp tục mở cho các tin nhắn tiếp theo.

Content-encoding: sẽ chỉ ra các loại mã hóa đang được sử dụng cho nội dung.

Content-lengt: chỉ định độ dài của nội dung thư.

Content-Type sẽ chỉ ra loại nội dung có trong thư chẳng hạn như text/html sẽ là tài liệu html.

Transfer-Encoding: chỉ định hình thức mã hoá được sử dụng để chuyển nội dung trọng tải đến người dùng một cách an toàn.

**Request Headers:**

Accept: cho máy chủ biết loại nội dung mà máy khách sẵn sàng chấp nhận, chẳng hạn như các loại hình ảnh, định dạng tài liệu văn phòng, v.v.

Accept-Encoding: cho máy chủ biết những loại nội dung mã hóa máy khách sẵn sàng chấp nhận.

Authorization: Được sử dụng để gửi thông tin xác thực user-agent cho server để truy cập các tài nguyên được bảo vệ.

Cookie: gửi cookie đến máy chủ đã được nhận trước đó.

Host: chỉ ra tên máy chủ có trên URL. Mục địch của heard này để chỉ ra trang web mình muốn truy cập khi mình gửi request tới một con server chứa nhiều trang web và header này giúp ta xác định được tên web mà ta muốn truy cập.

If-Modified-Since: sẽ chỉ ra thời điểm trình duyệt nhận được yêu cầu lần cuối. Nếu không có sự thay đổi server sẽ phản hồi lại 304. Header này chỉ hoạt động với GET và Head method.

If-None-Match: sẽ đưa ra một entity tag , trình duyệt sẽ gửi kèm theo entity tag được nhận từ server. Máy chủ có thể sử dụng entity tag để xác định xem trình duyệt có thể sử dụng bản sao tài nguyên được lưu trong bộ nhớ cache của nó.

Origin: sẽ chỉ ra nguồn gốc truy vấn từ đâu. Nội dung header bao gồm tên miền, giao thức , địa chỉ , port nếu có.

Referer: Địa chỉ của trang web trước đó mà từ đó có một liên kết đến trang được yêu cầu hiện tại.

User-Agent header được sử dụng để cung cấp thông tin về trình duyệt hoặc phần mềm tạo ra request.

**Response Headers:**

Access-Control-Allow-Origin: cho biết liệu tài nguyên có thể được được truy xuất thông qua các yêu cầu cross-domain Ajax.

Cache-control: Đưa ra các chỉ thị về bộ nhớ đếm cho trình duyệt. vd: cache-control: no-cache sẽ không có bộ nhớ cache.

Etag: là một entity tag. Client có thể gửi mã này yêu cầu tài nguyên trong tiêu đề if-None-match để thông báo cho server tài nguyên trình duyệt đang giữ trong bộ nhớ cache.

Expires: sẽ cung cấp thời gian của nội dung of phần tin nhắn body khá dụng. Trong khoảng thời gian đó trình duyệt có thể sử dụng bản sao lưu trong bộ nhớ cache.

Location: là header được sử dụng trong các redirect response để chi ra mục tiêu redirect.

Pragma: chuyển các chỉ thị của bộ nhớ cache cho trình duyệt.

Server: cung cấp thông tin về phần mềm máy chủ web đang xử dựng.

Set-cookie: sẽ trả về cookie cho trình duyệt để sử dụng cho các yêu cầu tiếp theo.

WWW-Authenticate: được sử dụng trong các response có mã 401 để cung cấp chi tiết về các loại xác thực mà máy chủ hỗ trợ.

X-Frame-Options: giúp ngăn chặn việc website bị chèn vào một website khác. Không cho phép chèn trang bằng các thẻ như <iframe>,<embed>...

**Cookies**

Cookie là một phần quan trọng của giao thức HTTP mà hầu hết các ứng dụng web dựa vào trên. Thông thường, chúng có thể được sử dụng như một phương tiện để khai thác các lỗ hổng. Các cơ chế cookie cho phép máy chủ gửi các mục dữ liệu đến máy khách, máy khách lưu trữ và gửi lại cho máy chủ. Không giống như các loại yêu cầu khác tham số (những tham số trong chuỗi truy vấn URL hoặc nội dung thư), cookie tiếp tục được gửi lại trong mỗi yêu cầu tiếp theo mà không có bất kỳ hành động do ứng dụng hoặc người dùng yêu cầu.

Server cung cấp cookie cho người dùng bằng header Set-Cookie trong response header.

Vd: Set-Cookie: tracking=tI8rk7joMx44S2Uu85nSWc.

Ở phía người dùng browser sẽ tự động thêm giá trị cookie đó vào header request tới server đã cấp cho mình cookie.

Vd: Cookie: tracking=tI8rk7joMx44S2Uu85nSWc.

Cooke thường bao gồm một một cặp name/value.Nhiều cookie có thể được cấp bằng cách sử dụng nhiều tiêu đề Set-cookie trong response của server. Chúng sẽ được gửi trở lại trong máy bằng cookie header, được phân tách giữa những cookie bằng dấu chấm phẩy.

**Cookie-flag**

Ngoài các giá trị của cookie, header Set-Cookie có thể bao gồm bất kỳ thuộc tính tùy chọn nào, được sử dụng kiểm soát trình duyệt xử lý cookie.

Expires: đặt ra giá trị và trình duyệt sẽ lưu cookie cho những lần tiếp theo cho đến khi hết hạn. Nếu không tồn tại giá trị, cookie chỉ được sử dụng cho phiên hiện tại.

Domain: để chỉ ra domain mà cookie này hợp lệ. Nó phải cùng giống nhau hoặc là parent của domain được nhận cookie.

path: đường dẫn chỉ định của URL path mà cookie hợp lệ.

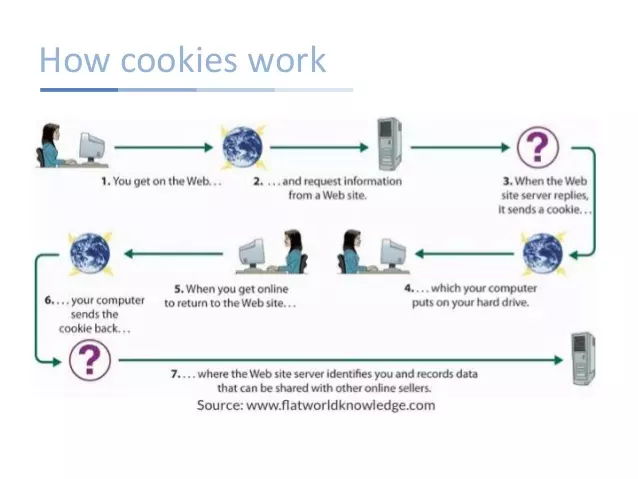
secure: giá trị sẽ được set nếu như đang thực hiện phiên truyền bảo mật.

vd: giao thức http thì secure sẽ không được set giá trị và https thì có vì https sử dụng cơ chế truyền bảo mật(mã hóa).

HttpOnly: thuộc tính này được đặt, cookie sẽ không thể truy cập trực tiếp qua javascript từ máy khách. Điều này giúp client không bị đánh cắp cookie từ phía ngoài.

Vd: người dùng click và một link thực thi một mã javascript nếu cờ này được set thì đoạn mã đó sẽ không được thực thi.

Cách hoạt động.



Khi bạn truy cập trang web từ trình duyệt và gửi một request đến server trước khi gửi request trình duyệt sẽ tìm trong máy file cookie(bạn có thể truy cập bằng cách nhấn Windows+r và gõ shell:cookies) sau khi tìm được giá trị mà trang web bạn cần đã thiết lập thì trình duyệt sẽ gửi dữ kiệu cookie của trang web đó tới trang web bạn truy cập. Nếu không tìm thấy nó sẽ không gửi.

Nếu người dùng yêu cầu truy cập trang nào đó và cookie đó hợp lệ thì máy sẽ cho phép người dùng đó truy cập. Trường hợp nếu không có cookie thì web sẽ coi đây là lần đầu tiên đăng nhập và sẽ tạo một giá trị cookie gửi cho trình duyệt của bạn để sử dụng cho những lần sau.

Question:Vì sao website có thể lưu phiên đăng nhập của người dùng.

Vì trình duyệt sẽ lưu cookie của người dùng và server sẽ lưu session của người dùng trên một file. Khi gửi request một tài nguyên nào đó trình duyệt sẽ gửi kèm cookie cho server nếu cookie này có giá trị xác thực với session có trong file thì server sẽ trả về phiên session cho người dùng.

VD: khi truy cập trang web bán hang và bạn log-in nó sẽ cho bạn một đoạn cookie và phiên đăng nhập. Khi này nếu bạn đóng trình duyệt và sau đó mở lại khi request lại trang web đó trình duyệt sẽ gửi lại cookie có chứa thông tin đăng nhập và sẽ trả lại session đó cho bạn mà không cần đăng nhập.

**Status Codes:**

Mỗi thông báo response http sẽ chứa status code trong dòng đầu tiên và cung cấp kết quả của request . Các mã được chia thành 5 nhóm phân loại bằng chữ số đầu tiên.

1xx-information:thông báo với client rằng server đã nhận được request.

2xx-Sucess: Thông báo rằng server đã xử lý thành công request

3xx-redirect: Thông báo rằng server đã chuyển request hiện tại sang request khác và client cần thực hiện gửi request tiếp theo để hoàn tất. Thông thường khi nhận được status code như này sẽ tự động gửi request tiếp theo để lấy kết quả.

4xx-Client\_error: Thông báo đã xảy ra lỗi từ phía clien trong khi gửi request.

5xx-server\_error: Thông báo đã xảy ra lỗi từ phía server trong khi xử lý request.

Sau đây là một số mã status phổ biến thường gặp trong lúc tấn công.

100 Continue: Chỉ một phần request được nhận bởi server. Lúc này client có thể gửi tiếp hoặc loại bỏ request.

200 OK: request đã được tiếp nhận và xử lý thành công.

201 Created: được trả về để thông báo rằng request đã thành công và chỉ hoạt động ở PUT request.

301 Moved Permanently: các request sẽ được chuyển đến một URL mới.

302 Found: giống như với status 301 nhưng URL lúc này chỉ bị chuyển hướng tạm thời. Do đó các URL gốc vẫn sẽ được sử dụng ở các request sau.

304 Not Modified: Code này được sử dụng cho mục đích caching. Nó cho client biết rằng có thể sử dụng bản sao được lưu trong bộ nhớ cache về tài nguyên mà người dùng cần yêu cầu. Server sử dụng header If-Modified-Since và If-NoneMatch để kiểm tra client có phiển bản mới nhất của tài nguyên đó chưa.

400 Bad request: server không thể xử lý request do bị sai dẫn đến server không thể hiểu và xử lý.

401 Unauthorized: được sử dụng để xác thực trước khi người dùng truy cập một số tài nguyên được bảo vệ.

403 Forbidden: xảy ra khi bạn truy cập một tài nguyên mà không có đủ quyền hạn.

404 Not found: tài nguyên này không tồn tại.

405 Method Not Allowed: xảy ra khi bạn sử dụng các method http mà url không được hỗ trợ.

413 Request Entity Too Large: xảy ra khi bạn gửi các yêu cầu với độ dài quá lớn khiến server không thể xử lý được.

414 Request URI Too Long: xảy ra khi client gửi một request với url quá dài khiến server không thể xử lý.

500 Internal Server Error: xảy ra khi server gặp phải các sự cố và không thể hiển trị được tài nguyên người dùng cần.

503 Service Unavailable: xảy ra khi máy chủ tạm thời ngưng hoạt động. Lỗi xảy ra có thể do máy chủ bận xử lý lượng lớn request dẫn đến không còn tài nguyên hoặc đang bảo trì.

1. Tìm hiểu cấu trúc của 1 URL

URL là viết tắt của Uniform Resource Locator, dịch sang tiếng Việt là định vị tài nguyên thống nhất. Cụ thể, URL là địa chỉ của một tài nguyên duy nhất trên Web. Mỗi URL hợp lệ sẽ trỏ đến một tài nguyên duy nhất, tài nguyên đó có thể là trang HTML, tài liệu CSS, hình ảnh, video, file PDF... Trong một số trường hợp ngoại lệ, URL có thể trỏ đến những tài nguyên không còn tồn tại hoặc đã bị di chuyển sang địa chỉ khác.

Cấu trúc: gồm những phần chính:scheme,subdomain,top-level-domain,port,second-level-domain,subdirectory.

VD: https://www.example.com:80/folder

Scheme: hiển thị ra giao thức mà server tiếp nhận khi ta truy cập trang web, như vd ở trên scheme của nó sẽ là giao thức https

Subdomain: nó chính là tên miền phụ của một trang web , nó xuất hiện ở trước tên miền chính như vd ở trên chính là www

Second-level-domain: Là tên website mà người dùng đặt cho nó.Như vd trên sẽ là example.

Top-level-domain: chỉ định loại miền mà bạn đã đăng ký. Như vd trên sẽ là com ngoài ra còn có một số loại khác như là: net,org,gov,edu,asia,vn, com.vn …

Port: ở đây chỉ số cổng đang sử dụng cho server ở vd trên là 80

Subdirectory: hay còn gọi là sub-folder nó như các folder tồn tại trên web và bạn truy cập trang sau đó truy cập vào các folder có trên trang web đó. Như vd trên subdirectory là folder.

Ngoài ra trong 1 url còn có các thành phần khác như query,fragment…

Query: nó như một tham số lấy dữ liệu từ người dùng và trả về các kết quả chính xác hơn mà người dùng cần thường bắt đầu bằng dấu ? .

Vd: example.com/?search=xe( ở đây trên trang example.com ta sẽ thực hiện tìm kiếm qua param search để tìm các kết quả là xe.

Fragment:

Đây là 1 đoạn giá trị có vai trò trương tự như là Query String nhưng nó cung cấp các hướng dẫn tùy chọn cho phía client.

1. Web Functionaly

Ngoài các giao thức truyền thông cốt lõi thì các ứng dụng web sử dụng nhiều công nghệ để cung cấp chức năng của chúng.

**Server-Side Functionality**

World Wide Web ban đầu chứa nội dung hoàn toàn tĩnh. Các trang web bao gồm các tài nguyên khác nhau, chẳng hạn như các trang HTML và hình ảnh,chỉ cần tải lên máy chủ web và gửi đến bất kỳ người dùng nào yêu cầu họ. Mỗi khi một tài nguyên cụ thể được yêu cầu, máy chủ sẽ phản hồi với cùng một nội dung.

Ngày nay một trang web được tạo bằng tài nguyên động. Khi người dùng request nó thì server sẽ trả về các kết quả khác nhau. Các nội dung động được tạo ra bởi các script hay các chương trình máy tính. Họ có nhiều đầu vào khác nhau và xử lý các input này đưa ra cá output khác nhau.

Khi người dùng request một tài nguyên nó còn kèm theo các tham số và các request http có thể gửi các tham số này qua 4 cách chính:

* trong cấu truy vấn của URL(param)
* Trong đường dẫn file của REST-style URL.
* Trong http cokie
* Trong các form được gửi qua method POST.

Ngoài các cách đó ra server có thể yêu cầu bất kỳ phần nào làm đầu vào cho quá trình xử lý chẳng hạn như có thể sử dụng User-Agent header để tạo các nội dung được tối ưu hóa cho trình duyệt đang sử dụng.

Giống như phần mềm máy tính nói chung, các ứng dụng web sử dụng nhiều loại công nghệ ở phía máy chủ để cung cấp chức năng của chúng:

* ngôn ngữ lập trình như PHP,VBScript...
* web application platform như là ASP.NET , JAVA.
* web server như là Apache, IIS.
* Database như là mysql, ms-sql...
* các thành phần back-end khác như filesystem,SOAP, directory service.

**The Java Platform**

Trong nhiều năm, Java platform Enterprise Edition là một tiêu chuẩn thực tế cho các ựng dụng doanh nghiệp quy mô lớn. Ban đầu được phát triển bởi Sun Microsystems và hiện thuộc sở hữu của Oracle, nó cung cấp cho mình các kiến trúc cân bằng tải và rất phù hợp với việc phát triển theo mo-dun và tái sử dụng mã.Do có lịch sử lâu đời và được áp dụng rộng rãi, nhiều development tools , máy chủ ứng dụng , các framework có sẵn để hỗ trợ các developer. Java platform có thể chạy trên các nền tảng như Windows,Linux,Solaris.

Descriptions của các ứng dụng web dựa trên java mà bạn có thể nhầm lẫn thuật ngữ:

* Enterprise Java Bean (EJB) là một thành phần phần mềm tương đối nặng đóng gói logic của một chức năng kinh doanh cụ thể trong ứng dụng. Các EJB nhằm giải quyết các thách thức kỹ thuật khác nhau mà các nhà phát triển ứng dụng phải giải quyết, chẳng hạn như tính toàn vẹn của giao dịch.
* Plain Old Java Object (POJO) là một đối tượng Java thông thường, khác biệt từ một đối tượng đặc biệt chẳng hạn như EJB. POJO thường được sử dụng để biểu thị các đối tượng do người dùng xác định, đơn giản hơn và nhẹ hơn nhiều hơn EJB và những người được sử dụng trong các khuôn khổ khác.
* Java Servlet là một đối tượng nằm trên máy chủ ứng dụng và nhận yêu cầu HTTP từ máy khách và trả về phản hồi HTTP. Việc triển khai Servlet có thể sử dụng nhiều giao diện để tạo điều kiện phát triển các ứng dụng hữu ích.
* A Java web container là một nền tảng hoặc công cụ cung cấp thời gian chạy môi trường cho các ứng dụng web dựa trên Java. Ví dụ về các vùng chứa web Java là Apache Tomcat, BEA WebLogic và JBoss.

Nhiều ứng dụng web Java sử dụng các thành phần mã nguồn mở và bên thứ ba cùng với mã được xây dựng tùy chỉnh. Đây là một lựa chọn hấp dẫn vì nó làm giảm nỗ lực phát triển và Java rất phù hợp với cách tiếp cận mô-đun này. Đây là một số ví dụ về các thành phần thường được sử dụng cho các chức năng ứng dụng chính:

* Authentication — JAAS, ACEGI
* Presentation layer — SiteMesh, Tapestry
* Database object relational mapping — Hibernate
* Logging — Log4J

**ASP.NET**

ASP.NET là framework application web của Microsoft và là đối thủ cạnh tranh trực tiếp sang Nền tảng Java.

ASP.NET sử dụng .NET Framework của Microsoft, cung cấp một tập hợp các API. Do đó ứng dụng ASP.NET có thể được viết bằng ngôn ngữ .NET nào như C# hoặc VB.NET.

**PHP**

được xuất hiện từ một dự án cá nhân về sau được phát triển mạnh trong việc phát triển web. Nó thường kết hợp với các công nghệ miễn phí được gọi là LAMP stack(linux là hệ điều hành, apache là web server, mysql là database và php là ngôn ngữ lập trình cho ứng dụng web).

Nhiều ứng dụng và thành phần mã nguồn mở được phát triển sử dụng php.

* Bulletin boards — PHPBB, PHP-Nuke
* Administrative front ends — PHPMyAdmin
* Web mail — SquirrelMail, IlohaMail
* Photo galleries — Gallery
* Shopping carts — osCommerce, ECW-Shop
* Wikis — MediaWiki, WakkaWikki

**Ruby on Rails**

Rails 1.0 được phát hành vào năm 2005, tập trung mạnh vào kiến trúc Model-View-Controller . Điểm mạnh chính của Rails là tốc độ có thể tạo các ứng dụng theo hướng dữ liệu hoàn chỉnh.

**SQL**

Ngôn ngữ truy vấn (SQL) được sử dụng để truy cập các data trong database như oracle , ms-sql , mysql. Phần lớn web ngày nay sử dụng database dựa trên sql làm nơi lưu trữ data.

Cơ sở dữ liệu quan hệ lưu trữ dữ liệu trong các bảng, mỗi bảng chứa mốt số hàng và cột. Mỗi cột đại diện cho một trường dữ liệu như 'name', 'id'... và từng hàng đương dương với các giá trị được gán cho một số hoặc tất cả các cột này.

SQL sử dụng các truy vấn để thực hiện nhiều tác vụ như đọc thêm xóa sửa.. vd để lấy tên người dùng qua id có thể thực hiện truy vấn sau: select name fom users where id='1' lúc này nó sẽ lấy ra tên của user có id là 1.

Ngoài ra để triển khai các chức năng một số ứng dụng web lấy input từ người dùng và thực hiện câu truy vấn. nếu quá trình này không được thực hiện đúng , kẻ tấn công có thể gửi input độc hại để can thiệp vào database.

**XML**

Extensible Markup Language(XML) là một đặc điểm kỹ thuật để mã hóa dữ liệu trong một dạng máy có thể đọc được. Giống như bất kỳ ngôn ngữ đánh dấu nào, định dạng XML phân tách tài liệu thành nội dung (là dữ liệu) và đánh dấu (chú thích dữ liệu).

Đánh dấu chủ yếu được thể hiện bằng cách sử dụng các thẻ, có thể là thẻ bắt đầu, kết thúc thẻ, hoặc thẻ phần tử trống:

Vd: <tagname>

</tagname>

<tagname />

Thẻ bắt đầu và thẻ kết thúc được ghép nối thành các phần tử và có thể đóng gói tài liệu nội dung hoặc các phần tử con:

<pet>ginger</pet>

<pets><dog>spot</dog><cat>paws</cat></pets>

Thẻ có thể bao gồm các thuộc tính, là các cặp tên / giá trị:

<data version=”2.1”><pets>...</pets></data>

XML có thể mở rộng cho phép các tên thể và thuộc tính tùy ý. Các tài liệu XMl sẽ bao gồm Document Type Definition (DTD) sẽ định nghĩa các

thẻ và các thuộc tính sử dụng trong tài liệu và các cách để chúng có thể kết hợp.

XML được sử dụng để mô tả dữ liệu dưới dạng text, nên hầu hết các phần mềm hay chương trình đều có thể đọc được XML Vì đó nên XML có thể được sử dụng để trao đổi dữ liệu giữa các trương chình.

**Web Services**

là sự tích hợp giữa hai máy tính giúp tương tác hiệu quả hơn. VD ứng dụng java có thể tương tác với các ứng dụng java , .NET , PHP... có thể nói webservice là cách thống nhất ngôn ngữ giao tiếp giữa 2 máy.

SOAP (là viết tắt của Simple Object Access Protocol) – giao thức truy cập đối tượng đơn giản: SOAP là một giao thức dựa trên XML đơn giản cho phép các ứng dụng trao đổi thông tin qua HTTP.

UDDI (Universal Description, Discovery and Integration): UDDI là một tiêu chuẩn dựa trên XML để mô tả, xuất bản và tìm kiếm các dịch vụ web.

WSDL (Web Services Description Language) – ngôn ngữ định nghĩa web service: WSDL là một ngôn ngữ dựa trên XML để mô tả các dịch vụ web và cách truy cập chúng.

**Client-Side Functionality**

Để ứng dụng phía máy chủ nhận thông tin đầu vào và hành động của người dùng và trình bày kết quả cho người dùng, nó cần cung cấp giao diện người dùng phía máy khách. Tại vì tất cả các ứng dụng web đều được truy cập thông qua trình duyệt web, các giao diện này đều có chung một lõi công nghệ. Tuy nhiên, những điều này đã được xây dựng dựa trên nhiều những cách thức đa dạng và những cách thức mà các ứng dụng tận dụng công nghệ phía máy khách đã tiếp tục phát triển nhanh chóng trong những năm gần đây.

**HTMl**

Công nghệ cốt lõi được sử dụng để xây dựng giao diện web là ngôn ngữ đánh dấu siêu văn bản (HTML). Giống như XML, HTML là một ngôn ngữ dựa trên thẻ được sử dụng để mô tả cấu trúc của tài liệu được hiển thị trong trình duyệt. Từ đơn giản của nó bắt đầu như một phương tiện cung cấp định dạng cơ bản cho tài liệu văn bản, HTML đã phát triển thành một ngôn ngữ phong phú và mạnh mẽ có thể được sử dụng để tạo giao diện người dùng phức tạp và chức năng cao.

XHTML là sự phát triển của HTML dựa trên XML và có đặc điểm kỹ thuật so với các phiên bản HTML cũ hơn. Một phần của động lực cho XHTML là cần phải tiến tới một tiêu chuẩn cứng nhắc hơn cho đánh dấu HTML để tránh các thỏa hiệp khác nhau và các vấn đề bảo mật có thể phát sinh khi các trình duyệt bắt buộc phải chấp nhận các dạng HTML ít nghiêm ngặt hơn.

**Hyperlinks:**

Là các đường liên kết qua một mục tiêu khác bằng cách người dùng submit nó và sẽ tự động chuyển tới liên kết đó.

VD:<a href=”?redir=/updates/update29.html”>What’s happening?</a>

Khi người dùng click vào, browser sẽ tạo ra request trỏ tới dướng dẫn được đính kèm như trên

GET /news/8/?redir=/updates/update29.html HTTP/1.1

Host: mdsec.net

**Forms**

Là nơi người dùng sẽ điền và gửi dữ liệu đến server thông qua các biểu mẫu. Thông thường sẽ sử dụng các html form để lấy đầu vào từ người dùng thông qua trình duyệt của họ. VD:

<form action=”/secure/login.php?app=quotations” method=”post”>

username: <input type=”text” name=”username”><br>

password: <input type=”password” name=”password”>

<input type=”hidden” name=”redir” value=”/secure/home.php”>

<input type=”submit” name=”submit” value=”log in”>

</form>

Form trên sẽ để người dùng điền vào user và password được gửi đến file xử lý là login.php.

Khi người dùng điền vào form và gửi trình duyệt sẽ tạo request sau

POST /secure/login.php?app=quotations HTTP/1.1

Host: wahh-app.com

Content-Type: application/x-www-form-urlencoded

Content-Length: 39

Cookie: SESS=GTnrpx2ss2tSWSnhXJGyG0LJ47MXRsjcFM6Bd

username=daf&password=foo&redir=/secure/home.php&submit=log+in

* Form này sử dụng method là Post nên trình duyệt sẽ tạo post request và đưa các dữ liệu vào body của request message.
* Sau đó là 2 tham số redir và submit được server sử dụng để xử lý logic của nó.
* URL của web chứa param là app, param này có thể được sử dụng để xử lý ở phía máy chủ.
* Có một param ở phần cookie là sess có thể được cấp từ phía máy chủ ở lần response trước.

**CSS**

Cascading Style Sheets (CSS) là một ngôn ngữ được sử dụng để mô tả bản trình bày của một tài liệu được viết bằng ngôn ngữ đánh dấu. Trong các ứng dụng web, nó được sử dụng để chỉ định cách hiển thị nội dung HTML trên màn hình.

CSS dựa trên các quy tắc định dạng có thể được xác định với các cấp độ khác nhau của tính cụ thể. Khi nhiều quy tắc khớp với một phần tử tài liệu riêng lẻ, các thuộc tính khác nhau được xác định trong các quy tắc đó có thể "phân tầng" thông qua các quy tắc này, do đó rằng sự kết hợp thích hợp của các thuộc tính kiểu được áp dụng cho phần tử.

Cú pháp CSS sử dụng các bộ chọn để xác định một lớp các phần tử đánh dấu mà một tập hợp các thuộc tính nhất định nên được áp dụng. Ví dụ, sau đây Quy tắc CSS xác định màu nền trước cho các tiêu đề được đánh dấu bằng cách sử dụng.

<h2> tags:

h2 { color: red; }

**JavaScript**

JavaScript là một ngôn ngữ lập trình tương đối đơn giản nhưng mạnh mẽ có thể dễ dàng sử dụng để mở rộng giao diện web theo những cách mà HTML không thể làm. Nó thường được sử dụng để thực hiện các tác vụ sau:

* Xác thực dữ liệu do người dùng nhập trước khi nó được gửi đến máy chủ để tránh yêu cầu không cần thiết nếu dữ liệu có lỗi.
* Tự động sửa đổi giao diện người dùng để đáp ứng các hành động.
* Truy vấn và cập nhật mô hình đối tượng tài liệu (DOM) trong trình duyệt để kiểm soát hành vi của trình duyệt (DOM của trình duyệt được mô tả trong một khoảnh khắc).

**VBScript**

nó tương tự như javascript nhưng chỉ hoạt động trên internet explorer. Được mô hình hóa trên Visual Basic và tương tác với Dom của browser.

**Document Object Model**

DOM là tên gọi viết tắt của (Document Object Model – tạm dịch Mô hình Các Đối tượng Tài liệu). Là một chuẩn được định nghĩa bởi W3C (Tổ Chức Web Toàn Cầu – World Wide Web Consortium). DOM được dùng để truy xuất và thao tác trên các tài liệu có cấu trúc dạng HTML hay XML bằng các ngôn ngữ lập trình thông dụng như Javascript, PHP…

**Ajax**

AJAX là chữ viết tắt của cụm từ Asynchronous Javascript and XML. AJAX là phương thức trao đổi dữ liệu với máy chủ và cập nhật một hay nhiều phần của trang web, hoàn toàn không reload lại toàn bộ trang.

Ajax được viết bằng Javascript chạy trên client, tức là mỗi browser sẽ chạy độc lập hoàn toàn không ảnh hưởng lẫn nhau. Về mặt kỹ thuật, nó đề cập đến việc sử dụng các đối tượng XmlHttpRequest để tương tác với một máy chủ web thông qua Javascript.

**JSON**

là một định dạng truyền dữ liệu đơn giản mà các ngôn ngữ lập trình ngày nay có thể hiểu được. Trong một tình huống điển hình, khi người dùng thực hiện một hành động, JavaScript phía máy khách sử dụng XMLHttpRequest để giao tiếp hành động với máy chủ. Máy chủ trả về một phản hồi nhẹ chứa dữ liệu ở định dạng JSON. Sau đó, tập lệnh phía máy khách sẽ xử lý dữ liệu này và cập nhật giao diện người dùng cho phù hợp.

Ví dụ, một ứng dụng web mail dựa trên Ajax có thể chứa một tính năng để hiển thị chi tiết của một số liên lạc đã chọn. Khi người dùng nhấp vào một liên hệ, trình duyệt sử dụng XMLHttpRequest để truy xuất chi tiết của liên hệ đã chọn, trả về bằng cách sử dụng JSON:

{

“name”: “Mike Kemp”,

“id”: “8041148671”,

“email”: “fkwitt@layerone.com”

}

Client-side script sử dụng trình phiên dịch Javascript để sử dụng JSON response và cập nhật các phần liên quan của giao diện user dựa trên nội dung của nó.

VD khi người dùng cập nhật thông tin chi tiết của một liên hệ, thông tin mới có thể được giao tiếp với máy chủ bằng cách sử dụng yêu cầu sau:

POST /contacts HTTP/1.0

Content-Type: application/x-www-form-urlencoded

Content-Length: 89

Contact={“name”:”Mike Kemp”,”id”:”8041148671”,”email”:”pikey@

clappymonkey.com”}

&submit=update6

**Same-Origin Policy**

là một chính sách quy định nội dung từ một website chỉ được đọc và thay đổi bởi một thành phần khác cùng site đấy, trường hợp truy cập nằm ngoài phạm vi site sẽ bị chặn.

Nếu không có SOP, khi người dùng vô tình truy cập một trang web độc hại, script được đặt sẵn trên này có thể truy cập được dữ liệu và thực hiện tính năng của bất kỳ trang web nào người dùng đã dùng trước đó. Chẳng hạn như thực hiện chuyển tiền, đọc mail hay chụp ảnh thẻ tín dụng khi giao dịch trực tuyến. Vì lý do này, các trình duyệt bắt buộc phải áp dụng SOP để ngăn chặn tương tác giữa các domain khác nhau.

Các lưu ý:

* Một trang nằm trên một miền có thể tạo ra một yêu cầu tùy ý đến một miền khác (ví dụ: bằng cách gửi biểu mẫu hoặc tải một hình ảnh). Nhưng bản thân nó không thể xử lý dữ liệu trả về từ yêu cầu đó.
* Một trang nằm trên một miền có thể tải một tập lệnh từ một miền khác và thực hiện điều này trong ngữ cảnh riêng của nó. Điều này là do các tập lệnh được giả định để chứa mã, chứ không phải dữ liệu, do đó, quyền truy cập tên miền chéo sẽ không dẫn đến tiết lộ bất kỳ thông tin nhạy cảm nào.
* Một trang nằm trên một miền không thể đọc hoặc sửa đổi cookie hoặc dữ liệu DOM khác thuộc về một miền khác.

**HTML5**

HTML5 là một bản cập nhật lớn cho tiêu chuẩn HTML. HTML5 hiện vẫn chưa phát triển và chỉ được thực hiện một phần trong các trình duyệt.

Từ góc độ bảo mật, HTML5 chủ yếu được quan tâm vì những lý do sau:

* Nó giới thiệu các thẻ, thuộc tính và API mới khác nhau có thể được tận dụng để cung cấp kịch bản trang web chéo và các cuộc tấn công khác.
* Nó sửa đổi công nghệ Ajax cốt lõi, XMLHttpRequest, để cho phép tương tác giữa các miền trong một số tình huống nhất định. Điều này có thể dẫn đến các cuộc tấn công tên miền chéo mới.
* Nó giới thiệu các cơ chế mới để lưu trữ dữ liệu phía máy khách, có thể dẫn đến các vấn đề về quyền riêng tư của người dùng và các loại tấn công mới như client-side SQL injection

**Browser Extension Technologies**

Vượt ra ngoài khả năng của JavaScript, một số ứng dụng web sử dụng các công nghệ browser extension sử dụng mã tùy chỉnh để mở rộng trình duyệt khả năng tích hợp sẵn theo những cách tùy ý. Các thành phần này có thể được triển khai dưới dạng bytecode được thực thi bởi một plug-in browser phù hợp hoặc có thể liên quan đến việc cài đặt các tệp thực thi gốc trên chính máy khách. Các công nghệ khách hàng dày dặn bạn có thể gặp phải khi tấn công các ứng dụng web.

* Java applets
* ActiveX controls
* Flash objects
* Silverlight objects

**State and Sessions**

Các công nghệ được mô tả cho đến nay cho phép các thành phần server và client của một ứng dụng web để trao đổi và xử lý dữ liệu theo nhiều cách. Tuy nhiên, để triển khai hầu hết các loại chức năng hữu ích, các ứng dụng cần theo dõi trạng thái tương tác của mỗi người dùng với ứng dụng qua nhiều yêu cầu. Vì ví dụ, một ứng dụng mua sắm có thể cho phép người dùng duyệt qua danh mục sản phẩm, thêm các mặt hàng vào giỏ hàng, xem và cập nhật nội dung giỏ hàng, tiến hành thanh toán và cung cấp chi tiết cá nhân và thanh toán.

Để làm cho loại chức năng này khả thi, ứng dụng phải duy trì một tập hợp dữ liệu trạng thái được tạo bởi hành động của người dùng qua một số yêu cầu. Cái này dữ liệu thường được giữ trong cấu trúc phía máy chủ được gọi là phiên. Khi một người dùng thực hiện một hành động, chẳng hạn như thêm một mặt hàng vào giỏ hàng của mình, ứng dụng bên máy chủ cập nhật các chi tiết liên quan trong phiên của người dùng. Khi mà người dùng sau đó xem nội dung trong giỏ hàng của mình, dữ liệu từ phiên được sử dụng để trả lại thông tin chính xác cho người dùng.

Trong một số ứng dụng, thông tin trạng thái được lưu trữ trên thành phần máy khách chứ không phải máy chủ. Tập hợp dữ liệu hiện tại được chuyển cho khách hàng trong mỗi phản hồi của máy chủ và được gửi trở lại máy chủ trong mỗi yêu cầu của khách hàng. Tất nhiên rồi, vì người dùng có thể sửa đổi bất kỳ dữ liệu nào được truyền qua thành phần máy khách, ứng dụng cần tự bảo vệ mình khỏi những kẻ tấn công có thể thay đổi điều này thông tin trạng thái nhằm cố gắng can thiệp vào logic của ứng dụng. Các Nền tảng ASP.NET sử dụng trường biểu mẫu ẩn được gọi là ViewState để lưu trữ nêu thông tin về giao diện web của người dùng và do đó giảm chi phí trên máy chủ. Theo mặc định, nội dung của ViewState bao gồm một hàm băm có khóa để ngăn chặn giả mạo.

Vì bản thân giao thức HTTP không có trạng thái, hầu hết các ứng dụng cần một cách để xác định lại người dùng cá nhân trên nhiều yêu cầu cho đúng tập hợp trạng thái dữ liệu được sử dụng để xử lý từng yêu cầu. Thông thường, điều này đạt được bằng cách phát hành mỗi người dùng một mã thông báo nhận dạng duy nhất phiên của người dùng đó. Những mã thông báo này có thể được truyền bằng cách sử dụng bất kỳ loại tham số yêu

cầu nào, nhưng hầu hết các ứng dụng sử dụng Cookie HTTP.

1. Encoding schemes

Các ứng dụng web sử dụng một số encoding schemes khác nhau cho dữ liệu của chúng.Cả hai giao thức HTTP và ngôn ngữ HTML trước đây dựa trên văn bản và các lược đồ mã hóa khác nhau đã được phát minh để đảm bảo rằng các cơ chế này có thể xử lý các ký tự bất thường và dữ liệu nhị phân một cách an toàn. Khi bạn đang tấn công một ứng dụng web, bạn sẽ thường xuyên cần mã hóa dữ liệu bằng cách sử dụng.

**URL encoding**

URL chỉ được phép chứa các ký tự có thể in được trong bộ ký tự US-ASCII - nghĩa là những bộ có mã ASCII nằm trong phạm vi 0x20 đến 0x7e,bao gồm. Hơn nữa, một số ký tự trong phạm vi này bị hạn chế vì chúng có ý nghĩa đặc biệt trong chính lược đồ URL hoặc trong HTTP giao thức.

Encoding schemes URL được sử dụng để mã hóa bất kỳ ký tự có vấn đề nào trong bộ ký tự ASCII mở rộng để chúng có thể được vận chuyển an toàn qua HTTP. Dạng mã hóa URL của bất kỳ ký tự nào là tiền tố% theo sau là mã ASCII hai chữ số của ký tự được biểu thị bằng hệ thập lục phân. Đây là một số các ký tự thường được mã hóa URL:

%3d — =

%25 — %

%20 — Space

%0a — Newline

%00 — Null byte

%20 — space

**Unicode Encoding**

Unicode là một tiêu chuẩn mã hóa ký tự được thiết kế để hỗ trợ tất cả các hệ thống chữ viết của thế giới. Nó sử dụng các lược đồ mã hóa khác nhau, một số có thể được sử dụng để đại diện cho các ký tự bất thường trong các ứng dụng web.

Mã hóa 16-bit Unicode hoạt động theo cách tương tự như mã hóa URL. Vì truyền qua HTTP, dạng ký tự được mã hóa Unicode 16 bit là tiền tố %u theo sau là điểm mã Unicode của ký tự được biểu thị bằng thập lục phân:

%u2215 — /

%u00e9 — é

UTF-8 là một tiêu chuẩn mã hóa có độ dài thay đổi sử dụng một hoặc nhiều byte để thể hiện từng ký tự. Để truyền qua HTTP, biểu mẫu được mã hóa UTF-8 của một ký tự nhiềubyte chỉ cần sử dụng mỗi byte được biểu thị bằng hệ thập lục phân và trước tiền tố %:

%c2%a9 — ©

%e2%89%a0 — z

Với mục đích tấn công các ứng dụng web, các bảng mã unicode có thể được sử dụng để bypass cơ chế xác thực đầu vào. Nếu filter chặn một số ký tự độc hại nhưng bộ xử lý lại hiểu được unicode , nó có thể bỏ qua bộ lọc bằng cách sử dụng unicode.

**HTML Encoding**

Mã hóa HTML được sử dụng để biểu diễn các ký tự có vấn đề để chúng có thể được kết hợp an toàn vào một tài liệu HTML.Các ký tự khác nhau có đặc biệt nghĩa là siêu ký tự trong HTML và được sử dụng để xác định cấu trúc hơn là nội dung của nó. Để sử dụng các ký tự này một cách an toàn như một phần của nội dung của tài liệu, cần phải mã hóa chúng bằng HTML.

Mã hóa HTML xác định nhiều entity HTML để đại diện cho ký tự cụ thể :

&quot; — "

&apos; — '

&amp; — &

&lt; — <

&gt; — >

Ngoài ra, bất kỳ ký tự nào cũng có thể được mã hóa HTML bằng mã ASCII của nó ở dạng thập phân:

&#34; — "

&#39; — '

hoặc bằng cách sử dụng mã ASCII của nó ở dạng thập lục phân (có tiền tố là x):

&#x22; — "

&#x27; — '

**Base64 Encoding**

base64 là phương thức convert dạng mã hóa 2 chiều từ binary sang string để có thể gửi đi được trong network một cách dễ dàng . Các binary lúc này sẽ được thể hiện bằng các ký tự mã ASCII . Nó thường được sử dụng để mã hóa các tệp đính kèm trong e-mail để truyền qua SMTP. Nó cũng được sử dụng để mã hóa thông tin đăng nhập của người dùng trong xác thực HTTP cơ bản.

Sự ra đời của base64 bắt nguồn từ việc mong muốn gửi một ảnh quả Email dùng chuẩn SMPT, tuy nhiên chuẩn SMTP chỉ cho phép chuyển các ký tự ASCII dùng 7 bit có giá trị từ 0- 127. Nhưng một tệp nhị phân bao gồm các byte có giá trị 0-255 vậy trước tiên chúng ta cần convert nó để có thể sử dụng được SMTP.

Quá trình thực hiện theo 4 bước.

* Dữ liệu nhị phân được sắp xếp theo từng khối 24 bit (3 byte) liên tục.
* Mỗi đoạn 24 bit được nhóm thành bốn phần 6 bit mỗi phần.
* Mỗi nhóm 6 bit được chuyển đổi thành các giá trị ký tự Base64 tương ứng của chúng. Mã hóa Base64 chuyển đổi ba octet thành bốn ký tự được mã hóa. (mỗi octet có 8bits dữ liệu).
* Người nhận sẽ phải đảo ngược quá trình này để khôi phục thông điệp ban đầu.

vd:ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/

Nếu khối dữ liệu đầu vào cuối cùng dẫn đến ít hơn ba phần đầu ra dữ liệu, đầu ra được đệm bằng một hoặc hai ký tự =.

Nhiều ứng dụng web sử dụng mã hóa Base64 để truyền dữ liệu nhị phân trong cookie và các tham số khác và thậm chí để làm xáo trộn dữ liệu nhạy cảm để ngăn các sửa đổi nhỏ.

**Hex Encoding**

Nhiều ứng dụng sử dụng mã hóa thập lục phân đơn giản khi truyền dữ liệu nhị phân, sử dụng các ký tự ASCII để đại diện cho khối thập lục phân.

Ví dụ: mã hóa hex tên người dùng “daf” trong cookie sẽ dẫn đến kết quả như này: 646166

Như với Base64, dữ liệu được mã hóa hex thường dễ phát hiện. Bạn nên luôn luôn cố gắng giải mã bất kỳ dữ liệu nào như vậy mà máy chủ gửi cho máy khách để hiểu chức năng của nó.

**Remoting and Serialization Frameworks**

Trong những năm gần đây các frameworks đã phát triển để tạo giao diện người dùng nơi mà client side code có thể truy cập từ xa các APi khác nhau. Điều này cho phép các nhà phát triển trừu tượng hóa một phần bản chất phân tán của các ứng dụng web và viết mã theo cách gần với mô hình của một ứng dụng máy tính để bàn thông thường. Các framework này sẽ cung cấp các stub API để sử dụng ở phía máy khách.Họ cũng tự động xử lý các lệnh gọi API này đến các chức năng phía máy chủ có liên quan và tuần tự hóa bất kỳ dữ liệu nào được chuyển đến các hàm đó.

VD:

* Flex and AMF
* Silverlight and WCF
* Java serialized objects

Question: Điều gì sẽ xảy ra nếu bạn nhập một địa chỉ vào thanh URL:

1. Đầu tiên bạn nhập địa chỉ example.com vào than hurl.
2. Trình duyệt sẽ kiểm tra dns trong cache để tìm địa chỉ ip tương ứng của domain vừa nhập.Nếu không có trong cache của trình duyệt nó sẽ tìm trong cache của hệ điều hành. Nếu không tìm được cache trong hệ điều hành nó sẽ tìm trong cache của router. Và cuối cùng nó sẽ tìm trên ISP cache
3. Nếu không tìm được DNS server của ISP sẽ tạo một truy vấn DNS để tìm địa chỉ IP của example.com. Đầu tiên nó sẽ gửi tới máy chủ root domain. Sau đó máy chủ root sẽ chuyển hướng đến máy chủ .com và sau đó sẽ chuyển hướng đến máy chủ example.com sau đó nó sẽ trả về IP của trang.
4. Khi tìm được IP nó sẽ khởi tạo kết nối TCP đến máy chủ tương ứng với địa chỉ vừa tìm được
5. Sau đó trình duyệt sẽ gửi HTTP request đến server
6. Server nhận được request và xử lý request đó và sau đó server sẽ trả về response kèm theo các status code cookie…
7. Khi nhận được response nếu không xảy ra lỗi hay yêu cầu xác thực … sẽ nhận được nội dung html. Và sau đó nó sẽ yêu các thẻ html các hình ảnh tệp css…Những tệp tĩnh sẽ được lưu lại ở cache và sẽ không cần request mới sau đó. Cuối cùng trang example.com sẽ hiển thị lên trình duyệt.