Security II - The Web Platform

Stefano Calzavara

Università Ca' Foscari Venezia

February 6, 2020

Università Ca'Foscari Venezia

Web Applications

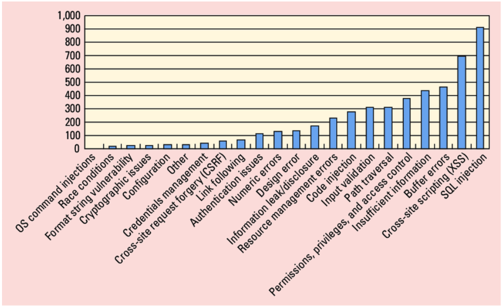
A web application is a client-server distributed application (normally) operated via a web browser:

- e-commerce sites: Alibaba, Amazon, Ebay
- mail services: Gmail, Outlook, Yahoo!
- social networks: Facebook, Instagram, Twitter

Fantastic tools which have become more and more popular over the years, yet extremely hard to protect!

Statistics

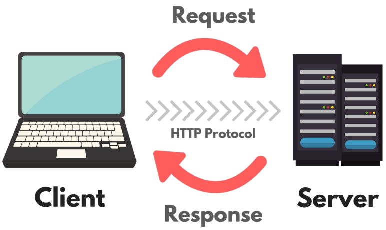
Trivia: how many web vulnerabilities in this plot?



The HTTP Protocol

The HTTP protocol is the workhorse protocol of the Web:

- simple request-response protocol in the client-server model
- plaintext: no confidentiality and integrity guarantees by default
- stateless: each HTTP request is handled as an independent event



Domain Names

On the Web, the server is typically identified by a string known as fully qualified domain name (FQDN).

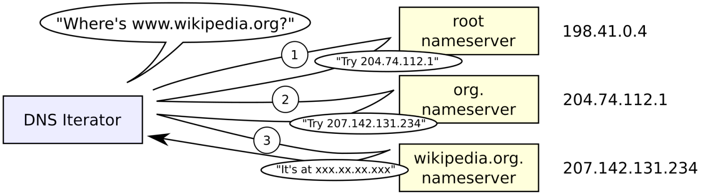
Terminology

The string `www.wikipedia.org` is a FQDN and:

- `www` is a hostname
- `wikipedia.org` is a domain name
- `wikipedia` is a sub-domain of the top-level domain `org`

Domain Name System (DNS)

The DNS system is used to resolve a FQDN into an IP address. There exists a many-to-many mapping between FQDNs and IP addresses.



Sometimes, the term hostname is abused and used interchangeably with FQDN in this context. Don't be confused!

HTTP Requests

HTTP requests are structured as follows:

1. a request line, including method, resource and protocol version
2. a list of request headers, including at least the Host header
3. an empty line, acting as separator
4. an optional request body

Example

```
POST /cart/add.php HTTP/1.1
Host: www.amazon.com

item=56214&quantity=1
```

HTTP Methods

The most common methods available in HTTP:

- GET: retrieves information from the server
- HEAD: like GET, but does not retrieve the response body
- POST: sends data to the server for processing
- PUT: uploads data (file) to the server
- DELETE: removes data (file) from the server
- OPTIONS: asks for the list of supported methods

HTTP Methods

Method	**Req. body**	**Resp. body**	**Safe**	**Idempotent**
GET	optional	yes	yes	yes
HEAD	optional	no	yes	yes
POST	yes	yes	no	no
PUT	yes	yes	no	yes
DELETE	optional	yes	no	yes
OPTIONS	optional	yes	yes	yes

First web security insight: remember this table, but do not trust it!

Query Strings

A query string is the part of the URL which assigns values to specified parameters. It is sent as part of the HTTP request.

Example

```
www.example.com/movies.php?id=54321&like=1
```

Query strings are particularly used to send data along with GET requests, but are not restricted to them.

HTTP Responses

HTTP responses are structured as follows:

1. a status line, including status code and reason message
2. a list of response headers
3. an empty line, acting as separator
4. an optional response body

Example

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8

<html><body>Done!</body></html>
```

HTTP Status Codes

Code	**Category**	**Example**
2XX	Success	200 OK
3XX	Redirection	301 Moved Permanently
4XX	Client error	401 Unauthorized
5XX	Server error	503 Service Unavailable

The only status codes with a clear semantics for web browsers are redirections, whose target is set in the `Location` header.

Response Body

The response body is just text. Yet, some text has special meaning to the browser.

HTML

Markup language used to define the structure of a web page:

- parsed as a DOM tree by the browser before rendering
- visually formatted for presentation by means of CSS

JavaScript

Client-side scripting language:

- full-fledged programming language, used basically on every website
- enriched with powerful APIs, which enable DOM manipulations

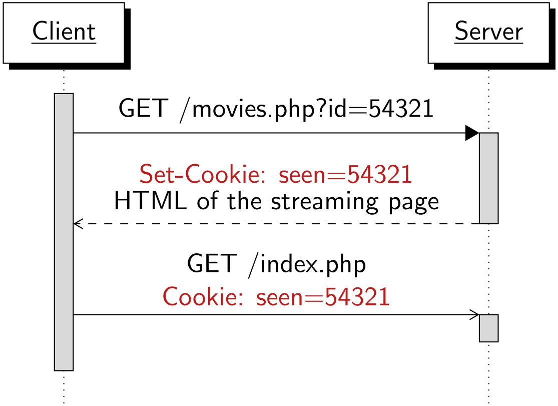
HTTP State Management

HTTP is a stateless protocol, yet state information can be stored at the client side by means of cookies:

- a cookie is a small piece of data of the form (key,value)
- set by the server into the client when desired
- sent by the client to the server along with HTTP requests

Cookies are opaque: the server can set whatever information inside the cookie in whatever format!

HTTP State Management



Domain Cookies

Web applications hosted on domains sharing a "sufficiently long" suffix[1] can also share cookies, using the `Domain` attribute. This can expose state information to multiple subdomains owned by the same organization.

Example

A web application at `accounts.example.com` can set a cookie with the `Domain` attribute set to `.example.com`, which is shared with all its sibling domains like `mail.example.com` (and their children).

[1] https://publicsuffix.org/

HTTP Secure (HTTPS)

HTTPS is the secure counterpart of HTTP:

- encrypted variant of HTTP based on the TLS protocol
- ensures confidentiality and integrity of the HTTP messages
- provides authentication of the server through signed certificates

Caveat!

HTTPS is necessary, but not sufficient, for web application security!

Attacking the Web

The most common question in security: what can go wrong?

Web Attacker

- owner of malicious website
- attacks via HTML and JS
- baseline attacker model

Network Attacker

- owner of the network
- full control of HTTP
- more and more important

Other attackers have been studied in the literature, but these are the most significant by far in most settings.