Security and Real World HTTP Servers

by sandeep chopra

LIGHTHOUSE LABS

learning agenda

Different HTTP server Security Issues

What’s the solution to those issues?

Quick overview of REST Convention

Recap Middleware

👋 Hands on Demo 🚀

SpaceX Launchpad Starter files

https://bit.ly/31x2WgR

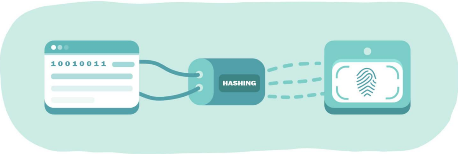
Security issue #1

Passwords save in plain text can be a huge security risk!

➔ It’s never a good idea to save any kind of credentials in plain text!

➔ Possible to read the content on Client storage, during Transfer and on Server side!

➔ What’s the solution then?



#hashing

What’s hashing?

- Hashing is simply passing some data through a formula that produces a result, called a hash.
- 3 Components of Hashing ->
  - Input Data e.g. text, document etc.
  - Hash Function e.g. the algorithm used to generate the fingerprint
  - Hash Value e.g. output of that hash function
- It’s unidirectional (one way)

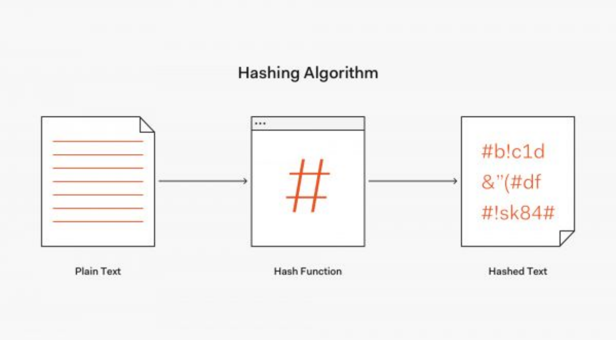
Good Hashing Function

a. Running the same hash function on the same input data must yield the same hash value.
b. Small changes to input data should result in large changes to the hash value.
c. Each resultant hash value for different input data should be unique.
d. The hashing process must be one way (i.e. it can’t be reversed).

MD-5

SHA-256

SHA-512



https://www.redcom.com/what-is-hashing/

Security issue #2

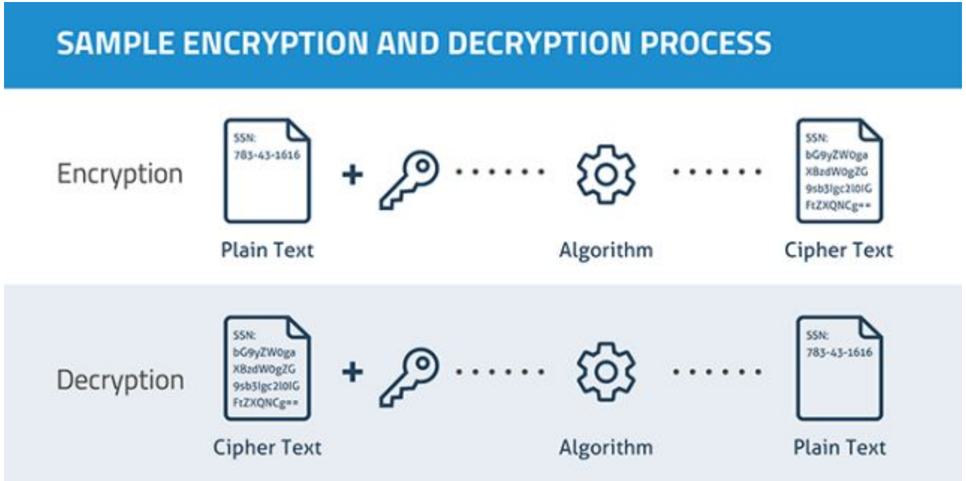
Cookies save in plain text can be a huge security risk!

- ➔ Our plain cookies are store in plain text which can be seen by anyone
- ➔ It is very easy to change the text of Cookie using dev tools or other means
- ➔ Hence very easy to hack someone's account or impersonate as another easy
- ➔ What's the solution?

#encrypting

What’s encryption?

- ➔ Encryption is the process of encoding/scrambling the date or information.
- ➔ It’s a two way process.



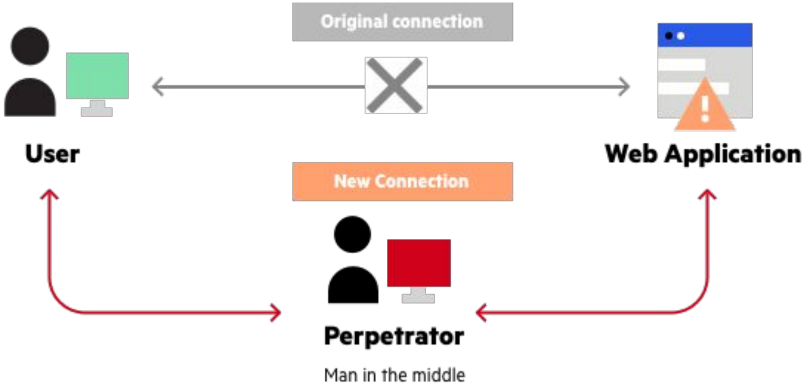
Security issue #3

Cookies can be stolen

- ➔ HTTP is plain-text protocol
- ➔ It’s easy to intercept HTTP calls which can lead to Man-in-the-middle attack
- ➔ What’s the solution?

#https

Man in the middle attack



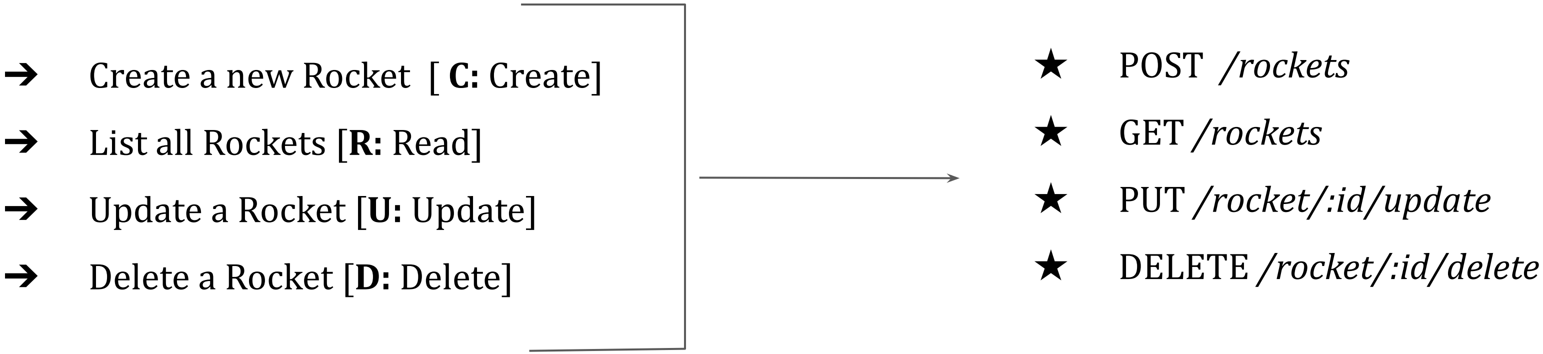
Restful API conventions

- REST is a pattern, a convention to organize our URL structure.
- Resource based routes convention.
- Resource id/information must be part of the url.
- It should use http verbs to express what the request wants to accomplish.

CRUD & REST conventions

REST stands for **Representational state transfer**

A map between HTTP verb/path combinations and CRUD actions users want to perform on a resource



➔ Create a new Rocket [**C:** Create]

➔ List all Rockets [**R:** Read]

➔ Update a Rocket [**U:** Update]

➔ Delete a Rocket [**D:** Delete]

★ POST */rockets*

★ GET */rockets*

★ PUT */rocket/:id/update*

★ DELETE */rocket/:id/delete*

Creating routes for our APP

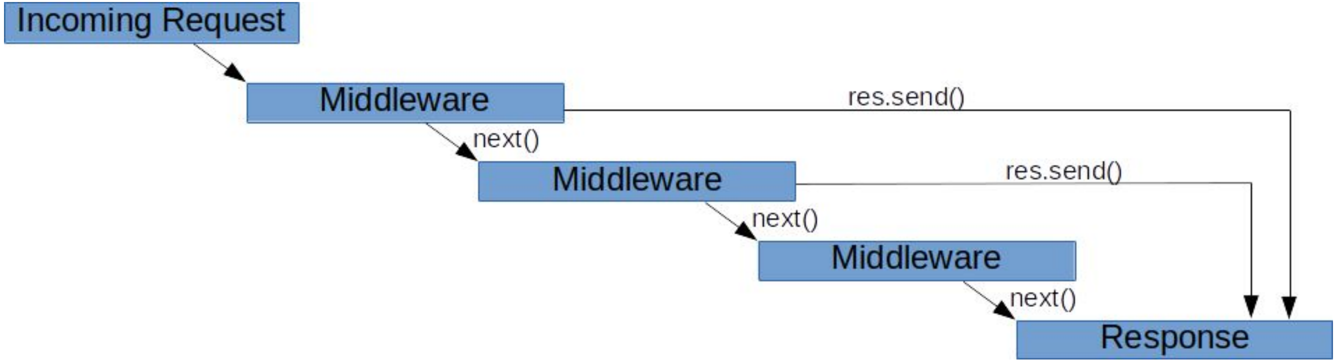
our **Resource** will be a single rocket 🚀

HTTP Verb	Route	Action
GET	/rockets	List all the rockets .
GET	/rockets/:id	Get a specific rocket.
GET	/rockets/new	Display the new rocket form.
POST	/rockets	Create a new rocket.
GET	/rockets/:id/update	Get the form to Update the existing rocket.
PUT	/rockets/:id	Update the existing
DELETE	/rockets/:id/delete	Delete a specific rocket

* browsers only support GET and POST, so our routes may vary slightly

Middlewares

- Middlewares helps us to extend the functionality of our server
- E.g. morgan, cookie-parser etc



SPACEX

Let’s Create Secret Dashboard

Implement login, logout functionality and protect
our routes from unauthorized users

Cookie-session to encrypt our cookie

Bcryptjs to hash our passwords

Questions?

👋 hands on practice