Identity on the Web

"Il mio nome è Remo Williams"

Why trace people?

- Businness optimization:
  - User profiling can aim to propose the "right" ads to a potentially buyer.
- Site optimization:
  - User profiling and surf history can aim to optimize the structure of the site; for example, if a class of users visit often and almost always a very deep zone of your site, you shoud consider to refactor the site in order to move up that zone.
- Surf optimization:
  - User profiling permits to personalize the site presentation.

How trace people?

How Web Surfing works (a fly-by):

- Page (HTML) fly over HTTP(S)
- HTTP is steteless (more or less) protocol
  - client shots (many times) and after server forget all.
- HTTP(S) fly over TCP protocol
  - TCP is a stateful protocol: this guarantees that pieces of the page are reassembled on the client in a correct way
- TCP fly thanks to IP addresses (and TCP ports)

client IP and its TCP ports can be traced naively (perhaps…)

Trace without help

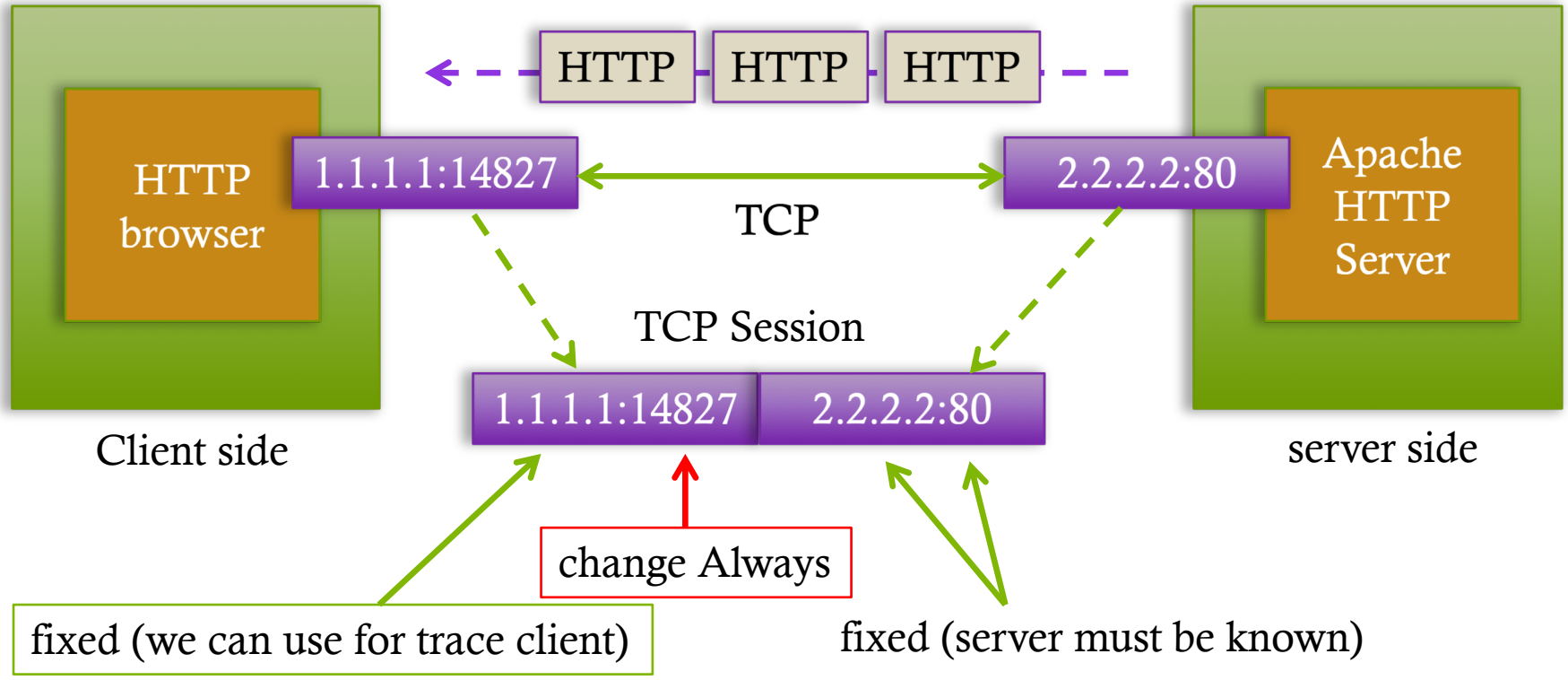
- We can (try to) trace client IP
  - Usually it change quikly, even more quick on mobiles
- Trace browser: User Agent on HTTP Header
  - When browser persent itself to the server, it give some informations about its capability, for example, Safari on the iPad has used the following:

```
Mozilla/5.0 (iPad; U; CPU OS 3_2_1 like Mac OS X; en-us)
AppleWebKit/531.21.10 (KHTML, like Gecko) Mobile/7B405
```

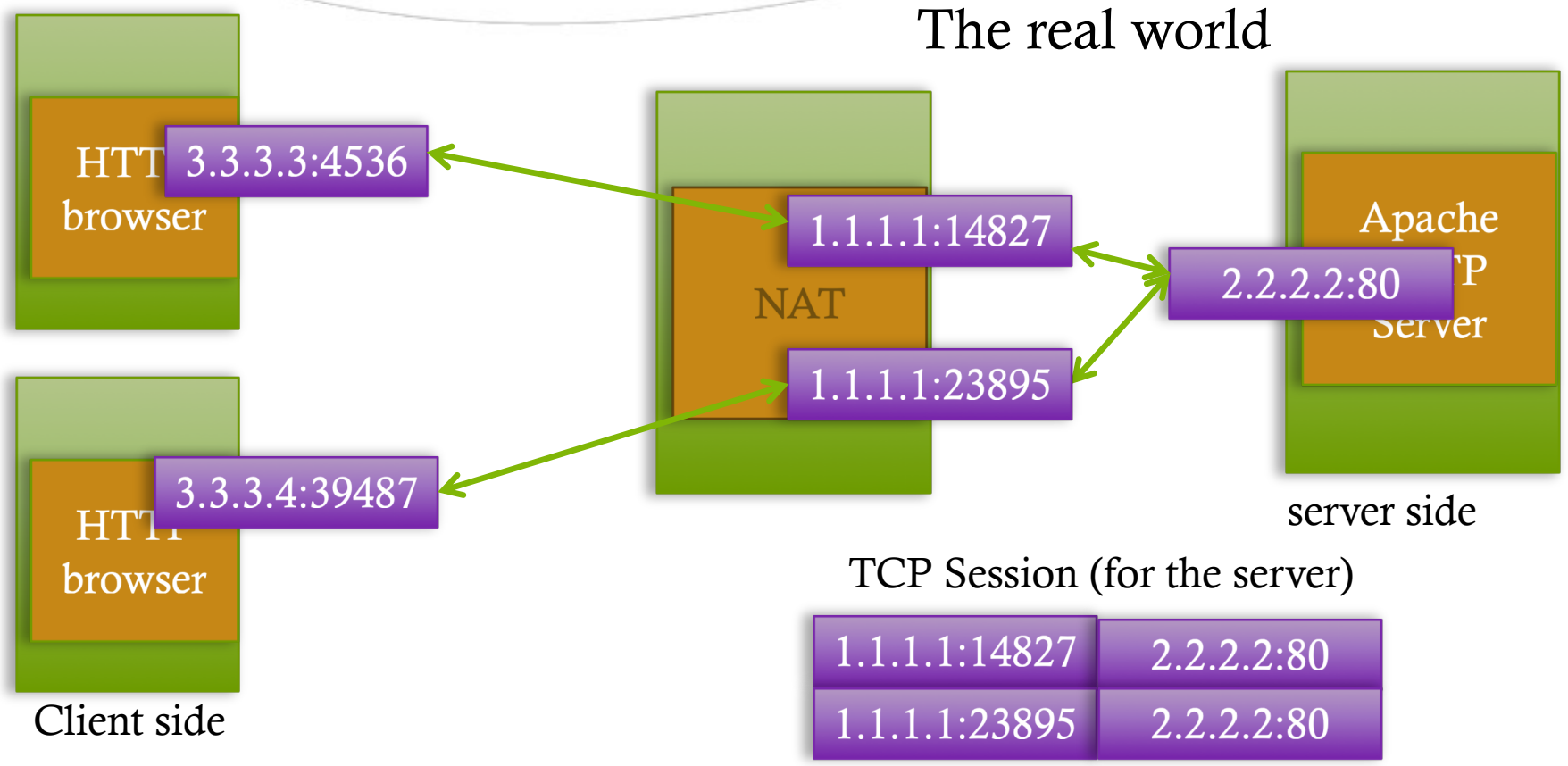
- https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers
- https://en.wikipedia.org/wiki/List_of_HTTP_header_fields

Transmit a HTTP Page

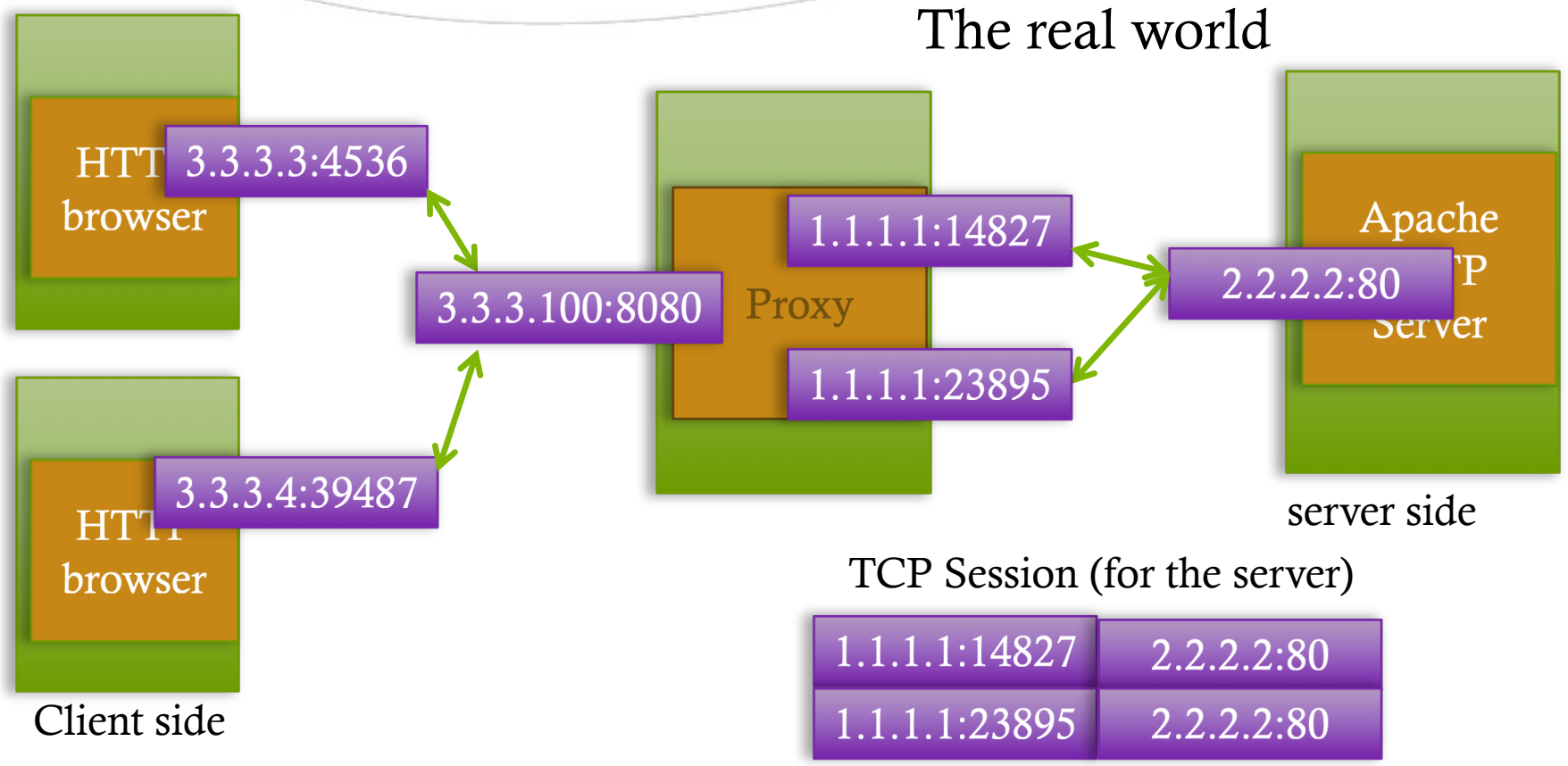
This beautiful world (in our dreams)



Client behind a NAT



Client behind a Proxy



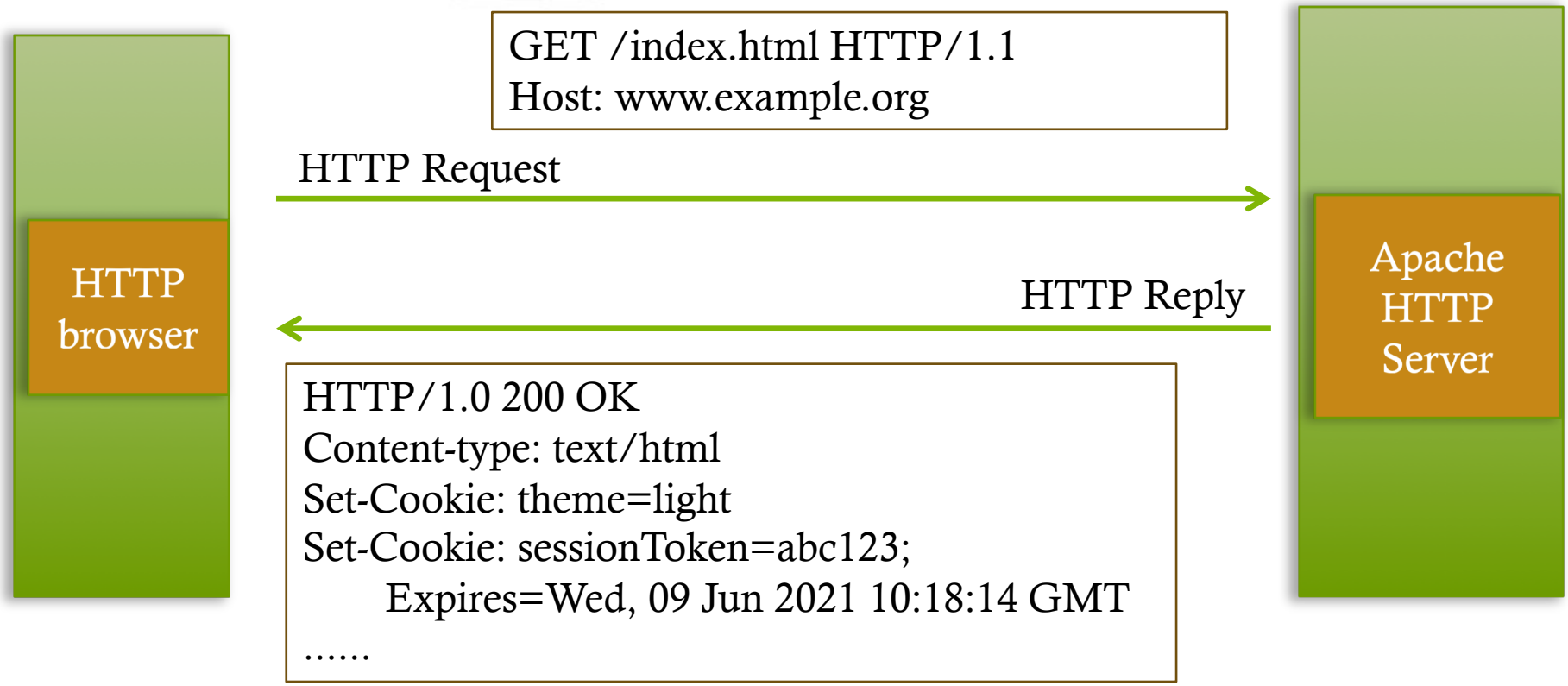
Trace with help: cookies

- Trace people without help is unfeasible
- A cookie is a small piece of text, sended by server and stored on a user's computer by their browser.

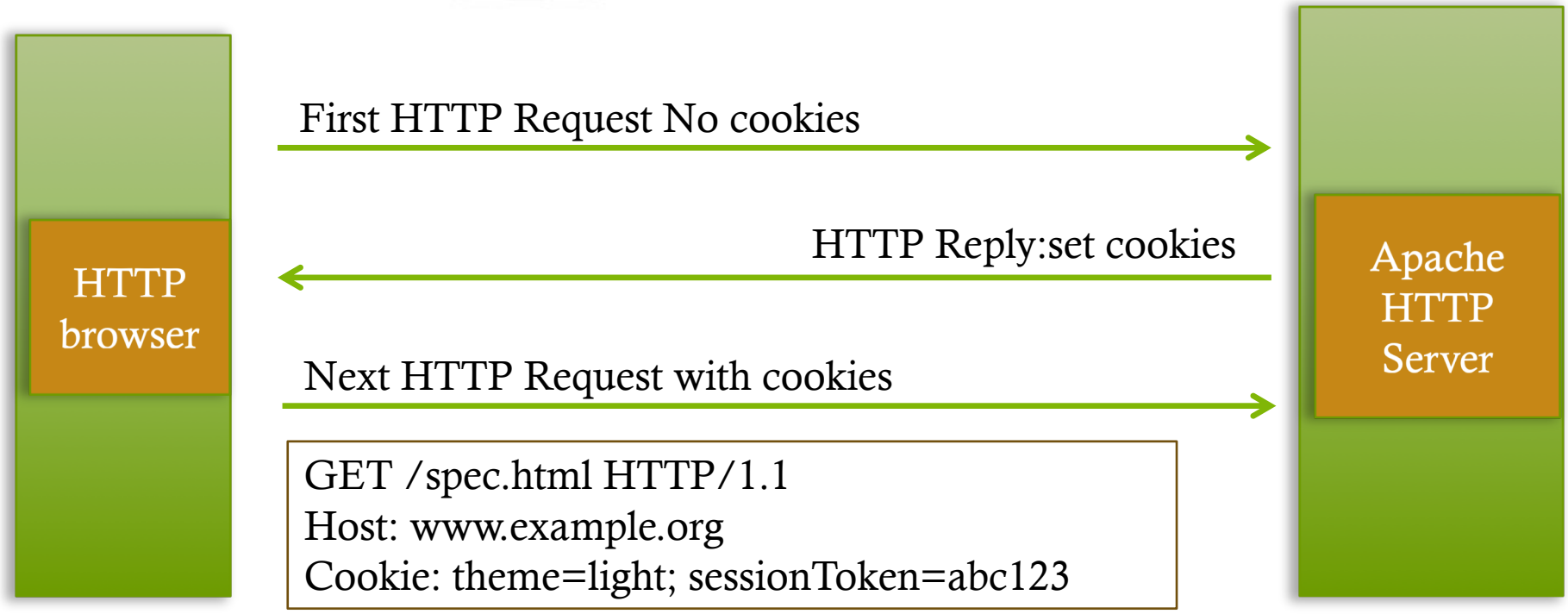
```
theme=light; sessionToken=abc123
```

- When client ask something to a server, cookies from that server are added to the header of the HTTP request.

Setting Cookies



Using Cookies



Cookie Types

- **Session cookie:**cookie die after close browser
- **Persistent cookie:** cookie die after fixed time
- **Secure cookie:** cookie is sent only on HTTPS
- **HttpOnly cookie:** cookie is not accessible to script
- **SameSite cookie:** cookie is sent only to originate site
- **Third-party cookie:** server send cokies for other sites
- **Supercookie:** cookie for top domains, such as .com
- **Zombie cookie: cookie that if deleted, then it is recreated**

  - https://en.wikipedia.org/wiki/HTTP_cookie

Cookies vs Identity

- Cookie permits to server to recognize a particular instance of borwser in a particular device during time; without other actions, the user identity is anonymous.
- If the client authenticates itself by some login process and the server associates this authentication to the understanding cookie, then the cookie became the proof of the user's identity and the protection of this *powerful* cookie became important.
- The **Secure**, **HttpOnly** and **SameSite** cookie types address variouse security concerns about maliciouse site which try to steal or misuse cookies.

EU cookie directive

- In 2002, the European Union launched the Directive on Privacy and Electronic Communications, a policy requiring end users'consent for the placement of cookies, and similar technologies for storing and accessing information on users' equipment.
- In particular, Article 5 Paragraph 3 mandates that storing data in a user's computer can only be done if the user is provided information about how this data is used, and the user is given the possibility of denying this storing operation.

Trust User Identity

"I am who I claim to be"

An identity problem

- Surfing on the web does not require particular knowledge about identities about clients and servers.
- Identity became a problem when some resources can be managed only by authorized users:
  - moving money from a bank account to another
  - watching private photos
  - reading news sites accessible only by subscription.

How can trust a site who claims to be the portal of my bank?
How my bank trust myself during my account operations?

Simmetric Key

- A symmetric key is a secret code knows by both of the two sides of a communication chanel.

- In examples, the two sides of the chanel are historically called "**A**lice" and "**B**ob"
  - Sometimes it needs to introduce a third actor, "**C**arl", often with the role of a robber/hacker/bad guy.
- Alice want to tell something really important and secret to Bob through this communication channel, but Bob want to be sure that Alice is really Alice before accept the message.
  - Carl usually wants to know or falsify the secret message, impersonates Alice and other such bad things.

Simmetric Key

Alice	Bob
"I'm Alice"	
	"Give me the proof!"
My secret key is "fuffi"	
	It match with ones I know ("fuffi")!, Welcome, Alice!'

Simmetric Key

- Some security concerns:
  - How securely share the secret shared key between Alice and Bob?
  - What's happens if someone (Carl) tap the key?
  - What's happens if Bob misuse the key?

- Some advantage:
  - Easy to implements
  - Cheap in term of cpu power consumption.

Carl steal secret key

Alice wants to speak with Bob on an insecure (HTTP) chanel.

Alice **Carl** **Bob**

"I'm Alice"

(....interesting...)

"Give me the proof!"

My secret key is "fuffi"

(I tap "fuffi"!!! Now I can became Alice!!!)

It match with ones I know ("fuffi")!,

Welcome, Alice!'

Carl impersonates Alice

Alice wants to speak with Bob on an insecure (HTTP) chanel.

Carl **Bob**

"I'm Alice"
(you cannot recognise me, heheheh)

"Give me the proof!"

My secret key is "fuffi"
(I tap the secret key!!! poor Alice!!)

It match with ones I know ("fuffi")!,
Welcome, Alice!'

Even Bob impersonates Alice

Dan is the Alice's banker, Bob want to steal money to Alice

Bob **Dan**

"I'm Alice"
(you cannot recognise me, heheheh)

"Give me the proof!"

My secret key is "fuffi"
(I dont need to tap the secret key, I know it!!! poor Alice!!)

It match with ones I know ("fuffi")!,
Welcome, Alice!'

I want to retrive all my money
(Alice's' money of course, heheheh)

Asymmetric Key

- To address the concerns on using simmetric key, we can use asymmetric keys.
- The asymmetric key is composed by two part:
  - The private key knows only by Alice,
  - The public key public sharable.
- With the symmetric key, It is possible to can sign *a token* with one part and verifing the sign with the other.

Asymmetric Key

Alice wants to speak with Bob on an insecure (HTTP) chanel.
The privat key of Alice is "3", the public keys is "6",

Alice **Bob**

"I'm Alice"

"Give me the proof!"

mumble... pow(pow(5,3),2)=15625
"I sign the stone '5' and the result '15625'"

mumble...radix(15625,6)=5
It match with "5"!,
Welcome, Alice!'

Certification Authority

"*Who's on the first base? Who's on the first base*!"

Alice on the air "I sign this stone with my (secret) sign"
Bob: "What is your (public) sign?"
Alice: "This is my (public) key! It's signed by (secret) sign of Carl"
Bob: "Ok, but what is the (public) sign of Carl?"
Alice: "This is the (public) key of Carl! It's signed by (secret) sign of Dan?"
Bob: "Oh Ok, I know Dan! good!"

- We need of a Trusted Cerfication Authority (CA)

Server Identity

- When a client connect a server, it may want to grant its identity
- SSL/TLS is a protocol to grants Identity by trust (a set of) CA, who is responsible to verify the identity claimed by a Server

Server: I give you my public key signed by CA

Client: I trust CA, ok.

Client: I cript (sign) "a strong secret word" with your public key, what is the secret word?

Server: I decript your secret word with my secret key, Its "a strong secret word"

Client: Ok, You are who you claim!

Server: And You??? Who you are?

Client Identity

- How trust client identity
  - SSL certificate, as server..... too complicate (and expensive)
  - A couple <user,password>, more better but if someone steal password?
  - two factor authentication: I authenticate the user twice, first by user password and after by some other shared secret:
    - phisical token
    - two way chanel communication
  - biometrical password (fingerprint, voice scan, face scan)