SDSLabs

Mini Lecture Series

Concise introductions to a wide array of tech and CS-related topics

HTTP vs HTTPS

A brief idea of how your data is **securely** transferred on the internet

Assumptions

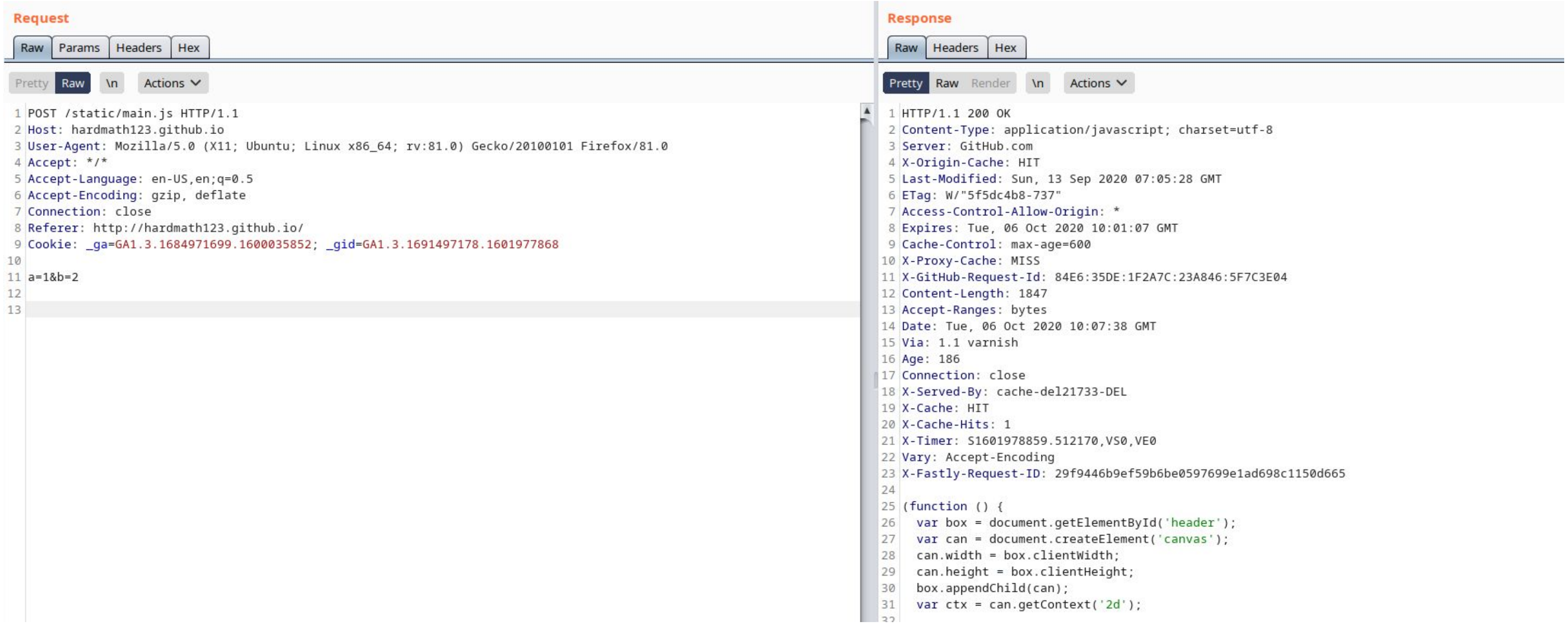
- 2 Types of Socket
  - Stream
  - Datagram
- We will concern ourselves with Stream Sockets only
- Every packet is safely received at the other end
- Ordering of Packets are reserved

Facts

- Protocol is a set of rules that the communicating parties agree upon for the packaging of data so that all transmissions can be interpreted properly at the other side. HTTP is one such protocol.
- If you are given a way of transmitting strings between computers, you can implement HTTP over it. It’s that easy*.

* Not that easy.

Anatomy of an HTTP Request



```
POST /static/main.js HTTP/1.1
Host: hardmath123.github.io
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:81.0) Gecko/20100101 Firefox/81.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Referer: http://hardmath123.github.io/
Cookie: _ga=GA1.3.1684971699.1600035852; _gid=GA1.3.1691497178.1601977868

a=1&b=2


```

```
HTTP/1.1 200 OK
Content-Type: application/javascript; charset=utf-8
Server: GitHub.com
X-Origin-Cache: HIT
Last-Modified: Sun, 13 Sep 2020 07:05:28 GMT
ETag: W/"5f5dc4b8-737"
Access-Control-Allow-Origin: *
Expires: Tue, 06 Oct 2020 10:01:07 GMT
Cache-Control: max-age=600
X-Proxy-Cache: MISS
X-GitHub-Request-Id: 84E6:35DE:1F2A7C:23A846:5F7C3E04
Content-Length: 1847
Accept-Ranges: bytes
Date: Tue, 06 Oct 2020 10:07:38 GMT
Via: 1.1 varnish
Age: 186
Connection: close
X-Served-By: cache-del21733-DEL
X-Cache: HIT
X-Cache-Hits: 1
X-Timer: S1601978859.512170,VS0,VE0
Vary: Accept-Encoding
X-Fastly-Request-ID: 29f9446b9ef59b6be0597699e1ad698c1150d665

(function () {
  var box = document.getElementById('header');
  var can = document.createElement('canvas');
  can.width = box.clientWidth;
  can.height = box.clientHeight;
  box.appendChild(can);
  var ctx = can.getContext('2d');
```

Routing

- ARP - Address Resolution Protocol
  - When someone asks for you, you raise your hand and you recieve your packet.
- IP Addresses - Are numerical labels that denote the virtual location of a device in a network

Man in the middle Attack

(✿◕‿◕) -------------(͡° ͜ʖ ͡°)---------------(͡⊙ ͜ʖ ͡⊙)

Alice Oscar Bob

Can we just use encryption to prevent this attack?

- If Alice and Bob know each other before hand, they can exchange a *secret key* before hand and can then easily encrypt their messages and prevent any intervention.
- But we don’t know everyone on the internet. So we need to devise a way of securely exchanging a common key with which we can communicate in form of encrypted messages.

Can we just use encryption to prevent this attack?

- Diffie-Hellman Key Exchange
  - A sends B public numbers g and p.
  - A and B generate a random but private number a and b respectively and calculate $g^b \bmod p$ and $g^b \bmod p$. They send each other these number publically.
  - Both of them raise the numbers received to the power of the secret they have and hence get the same secret key.

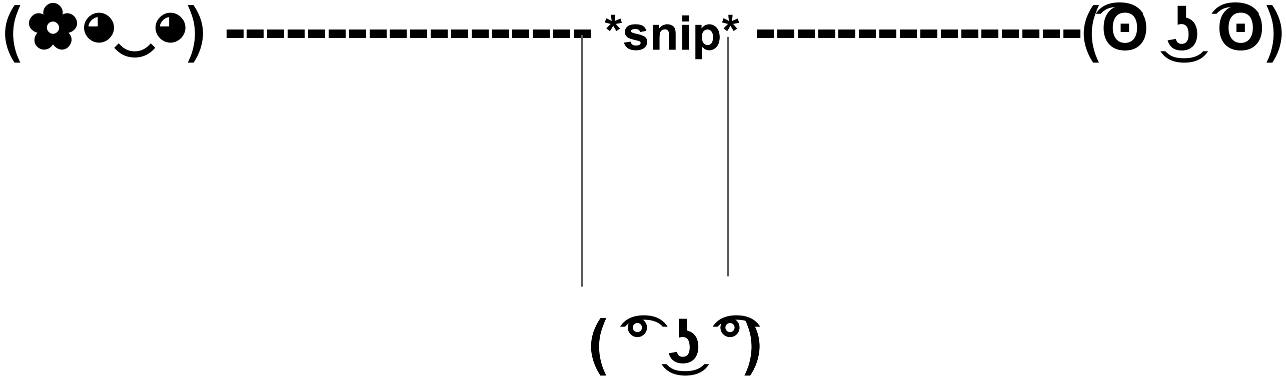
$$(g^b \bmod p)^a = (g^a \bmod p)^b = g^{ab} \bmod p$$

But wait ….

(✿◕‿◕) ----------------------------------(͡⊙ ͜ʖ ͡⊙)

(͡° ͜ʖ ͡°) ~~✂️~~

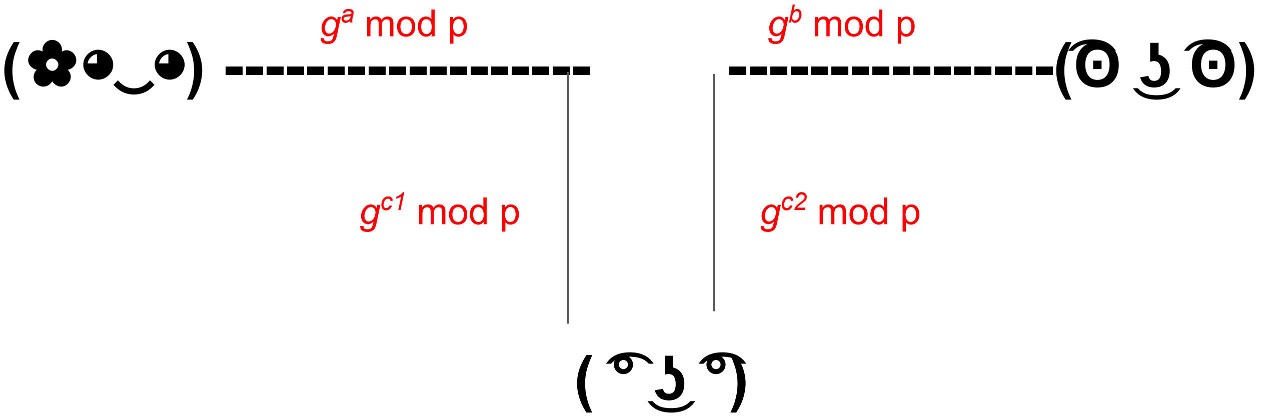
But wait ….



(✿◕‿◕) -------------------- *snip* -----------------(͡⊙ ͜ʖ ͡⊙)

(͡° ͜ʖ ͡°)

But wait ….



(✿◕‿◕)

g^a mod p

g^b mod p

(͡⊙ ͜ʖ ͡⊙)

g^{c1} mod p

g^{c2} mod p

(͡° ͜ʖ ͡°)

Can we just use encryption to prevent this attack?

Nope.

Digital Signatures?

- Signatures allow you to disclose a public key which would allow any third party to verify if a piece of information has been authenticated by you.
- The public key cannot be used to impersonate the original signer.

(✿◕‿◕)

I am Alice. Here is my public key. If you see any message sent on my name, check if it’s signed with my key. If the signature matches, it is indeed me.

(͡° ͜ʖ ͡°)

...

(͡° ͜ʖ ͡°)

Hi I am *Bob*. Here is my public key. If you see any message sent on my name, check if it’s signed with my key. If the signature matches, it is indeed me.

Digital Signatures?

Nope.

Establishing trust over the internet

A good way of knowing if you can trust someone would be if a mutually trusted third party could attest for their authenticity.

If a big trustworthy* corporation signed your name and public key, perhaps you could use this signature to prove your identity to someone when establishing trust.

SSL Certificates

SSL certificate for a website is a document containing

- Domain
- Issuer
- Domain owner
- Expiry date
- Public key of domain owner

The SSL certificate is signed by the issuer as proof of authenticity

(͡° ͜ʖ ͡°)

Hi I am *Bob*. Here is my public key. You can trust me because my own company OscarIndustries gave me this certificate. You can use their public key to verify the Certificate attached herein

Certificate Authorities (CA)

To prevent people from generated SSL certificates of their own, only certain organizations are authorized to issue valid SSL certificates. These are called CAs.

Who certifies an authority to be a CA? Another CA. This results in the formation of a long chain of trust which ends at a certificate which is trusted by your browser or PC. These certificates are called Root Certificates. (Cisco *wink wink*)

(✿◕‿◕) ---(͡ʘ ͜ʖ ͡ʘ)

(ಥ ͜ʖಥ)

Thanks for listening!

Here is an encrypted image of potato.