Total score: 85.5

Homework 1

Zhang Hexiao 20932780

1

a

$$d_{prop} = m/s$$

b

$$d_{trans} = L/R$$

c

$$m/s + L/R$$

−2 **d**

The last bit is at A and just about to transmit.

e

The first bit is on the link.

f

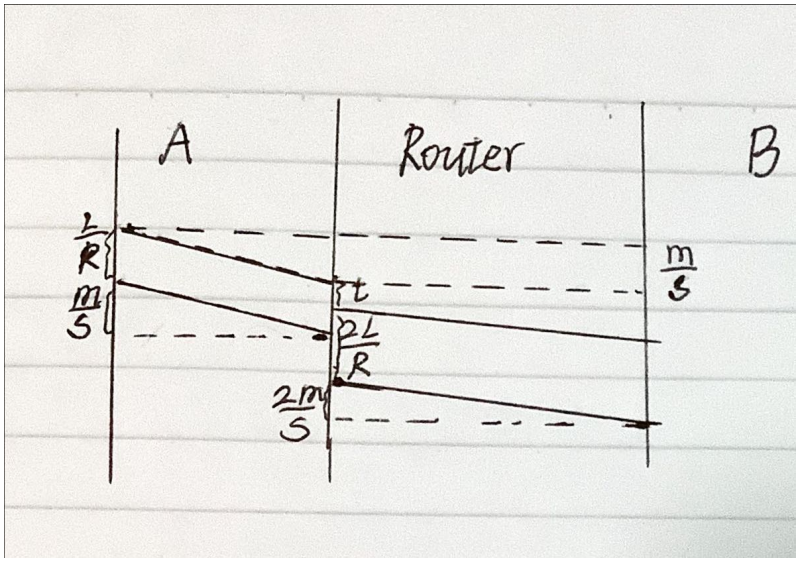
The first bit has reached at B.

−1 **g**

$$m = Ls/R = 62500m$$ pay attention to the unit

−4 **h**

$$3m/s + 2L/R + t$$



After receiving a packet (not a bit), the router processes it for t seconds before it forwards it.

2

a

$$1000\text{TB}/10Gbps = 8 \times 10^5 s \approx 9.26day > 1week$$

FedEx is a better choice.

b

$$100TB/10Gbps \approx 0.926day$$

Both FedEx and network can deliver in one day. So both are ok.

3

a

Yes for return code 200. The time was "28 Sep 2012 06:16:38 GMT".

b

31 Aug 2013 19:42:29 GMT

c

Yes.

On the one hand, HTTP/1.1 is stateless. Each request from the user is independent. On the other hand, the `Cache-Control` in the reponse is `no-cache`, which means the proxy server's cache is not used.

−3

d

> https://www.rfc-editor.org/rfc/rfc7231

what the cookie contains?

A cookie is a fingerprint of a user agent, which can include identifiers, expiry dates, the associated domain names, and so on.

−2

e

> https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Transfer-Encoding

When `Transfer-Encoding` is set as "chunked", data is sent in a series of chunks. The length of content is at the beginning of each chunk in hexadecimal format. The terminating chunk is a regular chunk with zero length.

when to use chunk?

So the size of this web object is 0x3fb0=16304.

−2.5

f

The first ten bytes are "<!DOCTYPE ".

missing chunk size

The server agreed to a persistent connection.

g

The `Vary` header describes the parts of the request message other than the method and URL that affect the content of the response.

For example, a client may have special needs for encoding. In this case, the server adds "Accept-Encoding" to the `Vary` header of the response, which tells the downstream proxy server how to match the headers of future requests to decide whether to use the cached response content or re-requesting.

h

> https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/ETag

The `ETag` header is an identifier for a specific resource version. It can be used to tell the downstream proxy server whether to use the cached response content, without resending the whole resourse.

If the resource at a given URL changes, a new `Etag` must be generated. A comparison of them can determine whether two representations of a resource are the same.

4

a

The AAAA record is the IPv6 version of the A record, matching a domain name to an IPv6 address.

The PTR record is used in reverse DNS lookups, likes `dig -x IP`, which gives the specific domain names with corresponding IP addresses.

b

DNS poisoning is using false DNS records to redirect users from a website to another, which can be applied to both end users and DNS servers. The forms include reply fake DNS records or direct tampering with user configuration.

DNSSEC can be helpful to avoid DNS poisoning, in which case a fingerprint (Hash) of DNS records is encrypted with a private key, and users use the public key to decrypt and verify it.

5

> https://www.rfc-editor.org/rfc/rfc7540
>
> https://datatracker.ietf.org/doc/rfc9114/

The disadvantages of HTTP/1.1 includes:

- Requests reusing: request pipelining can suffer from head-of-line blocking. Clients have to use multiple connections to achieve concurrency.
- High cost: the header fields are often repetitive and verbose.

HTTP/2 fixed these issues, which allows the interleaving of request and response messages on the same connection and uses more efficient header fields. It also provides the priority mechanism of request and enables more efficient processing of messages by binary message framing.

However, HTTP/2 is still constrained by TCP. If the packet is missed, the whole TCP connection suffers. What's more, TCP does not provide encryption and authentication. HTTP/3 addresses these problems through a new protocol QUIC based on UDP, providing better performance, lower packet loss rate, encryption, and authentication.

Compared with HTTP/1.1, HTTP/3 provides lower protocol overhead, multiplexing mechanism, lower packet loss rate as well as better security.