Exercise Session 3: Peeking under the Web - Solutions

COM-208: Computer Networks

The main goal of this exercise session is to get a sense of how web browsers and web servers communicate: we will play around with **caching** and **cookies**.

Caching at the web browser

Web browsers **cache** resources, so that they don't need to download them again if the user requests them again. You will now experience the difference this browser behavior can make.

Use a Firefox web browser, if you can. It comes with a nice tool, the web-developer network console (≡ → `More tools` → `Web Developer Tools` → `Network`), which visualizes each HTTP request that the browser makes, as well as the corresponding HTTP response that the browser receives. If you click on an HTTP request from the list on the left, you will see all the relevant information in the panel on the right.

Get ready to capture web traffic:

- Open your web browser and clear the cache. To do so in Firefox: ≡ → `Settings` → `Privacy & Security` → `Cookies and Site Data` → `Clear Data...`
- In Firefox, open the web-developer network console.
- Open Wireshark and start a new traffic capture.

The resources in this exercise are downloaded from web servers that use HTTPs instead of HTTP. HTTPs is a secure version of HTTP: HTTP messages encrypted within SSL (the Secure Sockets Layer that we mentioned in class) packets. This makes using Wireshark a bit harder, but we will guide you:

- To find out the web servers your web browser sent requests to, apply the filter: "`ssl and ssl.handshake.type==1`". This will display all the `Client Hello` messages. A `Client Hello` is the first packet that your computer sent to the web server to initiate their communication.
- To see the rest of packets exchanged with the web server: select the `Client Hello` message of that server, go to `Analyze` → `Follow` → `TCP Stream`. `Ignore/close` the window that pops up. Now you should see only the packets that belong to the same TCP connection as the packet you chose.

Answer the following questions, using the web-developer network console, or Wireshark, or (ideally) both.

- Visit Welcome to Rio. Where is this resource downloaded from?

From an EPFL web server. Using Wireshark, we see that the IP address of the web server is 128.178.32.48 (e.g., check the first SSL packet of type `Client Hello`). By running "`host 128.178.32.48`" from the terminal we see that 128.178.32.48 maps to DNS name `moodle.epfl.ch`.

- How long did it take to download it?

From the network console, we see that it took 129ms (This value can be different from one machine to another).

- Restart your web browser. Visit Welcome to Rio again. How long did it take to load it this time? What explains the difference?

This time the image loaded faster. The reason is that the web browser had cached it, and it did not have to retrieve it again (just to check that its cached copy was fresh).

- Open a second tab in your web browser and visit Welcome to Rio II. Where is this resource downloaded from? How long did it take to download it?

From a web server in the US: From Wireshark, we see that the IP address of the web server is 185.15.58.240 (e.g., check the first SSL packet of type `Client Hello`). By using an IP geolocation tool (e.g., https://tools.keycdn.com/geo) we see that the web server with IP 185.15.58.240 is in the US.

From the network console, we see that it took 161ms.

- When you visited ‘Welcome to Rio II’, you viewed the same picture as when you visited ‘Welcome to Rio’. Knowing that your web browser had already cached ‘Welcome to Rio’: do you think your browser served ‘Welcome to Rio II’ from the cache, or it downloaded it from its origin web server? Why do you think your browser behaved this way?

The web browser downloaded the image from its origin web server. To confirm this, we visit Welcome to Rio II again. We notice that it load faster the second time. This means, the resource has just been cached.

Web browsers identify images (and web objects in general) by URL. Given that the URLs of the two images are different, the web browser has no way of knowing that the content of the two images is the same, hence does not serve the second image from the cache.

Caching at a proxy web server

It is not only web browsers that cache resources; **proxy web servers** are web servers that act as **intermediaries**: they cache resources that are originally stored in other web servers (called **origin web servers**) and serve them to nearby web clients.

Before you start, clear your browser cache and find the proxy settings of your web browser. In Firefox: ≡ → `Settings` → `General` → `Network Settings` → `Settings`. Setup a proxy web server using the following settings:

- Check "`Manual proxy configuration`".
- Set `HTTP Proxy`: 167.172.238.15 and `Port`: 10002 (you could use any proxy web server, that does HTTPS caching, from https://free-proxy-list.net/).
- Check "`Also use this proxy for HTTPS`".

Now visit the same two resources that you visited before.

- Where were the resources downloaded from?

> Both resources were downloaded from the proxy web server: From Wireshark, we see that the IP address of the web server is 167.172.238.15 (e.g., check the SSL packets of type `Client Hello`).

- How long did it take to download each resource this time? Why did the download time change?

> From the network console, we see that it took 7335ms and 858ms to download 'Welcome to Rio' and 'Welcome to Rio II', respectively. These times are longer compared to not using the proxy.
>
> This is because the computer does not contact the server directly. It rather contact the proxy which in turn contacts the server. This is adds communication overhead.
>
> Furthermore, the choice of the proxy can result with even longer delays: if the proxy is further away and/or more busy than the origin web server.

- What benefit would a network (e.g., EPFL network) gain by installing a proxy web server?

Proxy web servers can act as a shared cache that stores responses shared among multiple users. A good reference about HTTP caching is https://developer.mozilla.org/en-US/docs/Web/HTTP/Caching.

- What will happen to your web browser if the proxy web server that you specified fails? Will your browser be able to load any mew web pages? What about pages that are cached?

Regarding new web pages, any attempt to load such a page will fail as traffic goes through the proxy. Regarding web pages that the browser has cached, the ability of the browser to load such a page depends on the freshness of the page (calculated based on several fields of the HTTP header): if the page is fresh, the browser will load it; if the page is stale, the browser will not be able to load it as the browser has to check with the proxy if the page has been modified (and does not receive a reply from the failed proxy).

IMPORTANT: Restore your original proxy settings: ≡ → `Settings` → `General` → `Network Settings` → `Settings` → `Use system proxy settings`.

Cookies

Cookies enable a web server to **link subsequent HTTP requests** to the same web browser: if you send 10 HTTP GET requests, for 10 different resources, to the same web server, the web server can use cookies to figure out that these 10 requests came from the same web browser, even if you did not explicitly provide any identification information (e.g., you did not login).

Before you start, figure out how to control cookie settings in your browser. In Firefox:

- To view or delete the cookies that have been stored on your computer: $\equiv \rightarrow$ `Settings` $\rightarrow$ `Privacy & Security` $\rightarrow$ `Cookies and Site Data` $\rightarrow$ `Manage Data` or `Clear Data...`
- You can also view the cookies that your computer sends along with an HTTP request, or receives along with the corresponding response, through the web developer network console: select an HTTP request from the list of requests on the left, then select the `Cookies` menu from the panel on the right.
- To view all the cookies stored due to visiting the web page: Go to `Storage` from the top panel, and then select `Cookies`.

Let us see cookies in action:

- Allow your browser to exchange cookies. Delete existing cookies. Open Moodle. Did the EPFL web server send you any cookies? And are they all from the same domain?

Yes, there are four cookies from two domains: two cookies from `.epfl.ch` and two from domain `moodle.epfl.ch`.

You can read more about EPFL's cookies and how they are used here.

- Login to your moodle account. Restart your web browser and re-open Moodle. Does it ask you to login again? Explain your browser's behavior.

No, we are automatically logged in. This happens because, along with the current HTTP request, our web browser sent a `TequilaPHP` cookie for `moodle.epfl.ch`, which contains our login information. This way the server "remembers" our credentials and directly shows the courses we are registered in.

You can read more about EPFL's Tequila authentication service here

- Delete existing cookies. Restart your web browser and re-open Moodle. Does it ask you to login again? Explain your browser's behavior.

Yes, it asks us to login again: since we deleted the cookies, the `TequilaPHP` cookie is longer there. So it is as if we visited the website for the first time.