CMSC 23200 HW 5

Ruolin "Lynn" Zheng

February 14 2020

Code File

1. assignment5-2.sql - the SQL create table command used for Problem 2
2. assignment5-2.html - the HTML file with placeholders, used for Problem 2 only
3. assignment5-2.php - the PHP file used for Problem 2's schema (as will be explained below, I updated my schema from Problem 2 to Problem 3)
4. track.sql - the SQL file containing commands to create the table for Problem 3 and 4 and queries to group together the visits
5. track.html - the HTML file used for Problem 3 and 4
6. track.php - the PHP file used for Problem 3 and 4

Problem 1

I designed my schema according to the eight sub-problems in Problem 3. The columns of my final schema are as follows: (For ease of development, I used VARCHAR instead of BOOLEAN some fields as described below.)

```
(
    id INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    page VARCHAR(255),
    cookie VARCHAR(255),
    ip VARCHAR(255),
    window_width INT,
    window_height INT,
    user_agent VARCHAR(255),
    enable_cookie VARCHAR(255),
    do_not_track VARCHAR(255),
    enable_popup VARCHAR(255),
    fonts VARCHAR(255)
)
```

`id` is just a unique identifier of an entry. `page` records the name of the .html file visited. `cookie` records the cookie if the client has `enable_cookie` set to `true`, else it will be an empty string. `ip` records the IP of the client. `window_width` and `window_height` together gives the browser window size as used in Problem 3's window-size sub-problem. `enable_cookie` is a boolean value that records whether the client enables cookies. `do_not_track` stores a string value instead of a boolean since `Navigator.doNotTrack` results in `null` in Chrome and is by default `unspecified` instead of `false` in Firefox. `fonts` is a string of length 5 consisting of 1's and 0's. It correspond to the five fonts in the order of 'Abyssinica SIL', 'DejaVu Sans', 'GFS

Baskerville', 'Liberation Sans', 'Roboto'. **1** indicates that the font is installed on the client's machine, and **0** indicates that the font is not installed. (In the schema for Problem 2, I stored one boolean value for each of the font. I updated my schema when working on Problem 3 to the current version to facilitate easier writes and queries.)

Problem 2

I created my table with the commands in **assignment5-2.sql**. My **assignment5-2.php** file will parse the values (cookie, IP, etc.) from a GET request, connect to the MySQL database, and insert the values into corresponding columns in the table. **assignment5-2.html** contains placeholder values (cookie, IP, ect.), upon loading the page, will direct the values to **assignment5-2.php** via a GET request.

Problem 3

0.1 Cookie

1. My site's cookie's key is `tracking_cookie` and its value is a randomly generated string. I retrieve the cookie if `tracking_cookie` is present in `document.cookie` and if it's not present, I assign the client a new cookie by setting the key-value pair `tracking_cookie` and a random string.

2. Cookie tracking can be stopped if clients disable cookies or regularly delete cookies. Cookies, especially ones with long expiration days, are perhaps the most straightforward way for a server to uniquely identify a client. Therefore, eliminating cookies will make tracking and device fingerprinting a lot harder. However, I probably wouldn't eliminate this feature because cookies usually store information to lend more convenience to our browsing. (For example, using just cookies, some sites might be able to remember my preference without me having to sign up for an account and stay logged in.)

0.2 IP

1. I used JQuery to issue an IP address lookup via the ipify API (https://www.ipify.org/).

2. It is difficult to stop IP tracking by eliminating browser features as geolocation is a physical feature. However, clients could possibly stop or thwart IP tracking by connecting over an VPN. Without IP tracking, websites cannot easily map out their clients' geolocation for displaying relevant Ads, news, and so on. (Websites associated with politics might also have the incentive to access geolocation information.) As a feature, IP addresses cannot be eliminated, but connecting over VPNs would provide some protection for the clients' privacy.

0.3 Window Size

1. I directly access `window.innerWidth` and `window.innerHeight` in JavaScript.

2. This type of tracking can be eliminated if client-side JavaScript is prevented from accessing the height and width properties of the browser window. Without window size tracking, websites will lose an easy way of telling apart visits from devices with drastically different screen sizes (phones, tablets, laptops, even external displays). (Note this might be under the assumption that most devices use browsers full-screen. When I have my developer console open when browsing the website, the window size changes.) Despite that, I wouldn't eliminate this feature since window size offers useful information about what kind of display and HTML rendering is most suitable for a specific device.

0.4 User Agent

1. I directly access `navigator.UserAgent` in JavaScript.

2. User agent plays a role similar to window size. It can be eliminated only if client-side JavaScript is no longer allowed to access said property. Without it, websites won't be able to tell the client's device model, operating system, browser type, version, and so on. Although this may be a key distinguishing feature in device fingerprinting, I wouldn't disable this feature because this information will lend flexibility to HTML rendering in the browser in the case of browser-incompatible content (because of different browser type or version). The argument for window size also applies.

0.5 Enable Cookies

1. I directly access `navigator.cookieEnabled` in JavaScript.

2. This feature can be eliminated only if JavaScript is prevented from learning whether the browser enables cookie. This might create a small inconvenience for websites: Suppose a client has disabled cookies. A website won't be able to tell and will repeatedly assign new cookies to the client on their every visit. If the website were relying solely on cookies for tracking, they might mistake the visits from this one single client as from multiple different clients. Since the server needs to know whether the client has enabled cookies, it doesn't make much sense to eliminate this feature.

0.6 DNT

1. I directly accessed `navigator.doNotTrack` in JavaScript.

2. This is a tricky one: A flag indicating that the client doesn't want to be tracked could actually be a distinguishing feature since not many people have this turned on nor do many browsers beside Firefox support it. This is a valuable piece of information especially in the case where a website knows little of the other features. A client with DNT on must be someone who has some sort of a security mindset. Since we discussed in lecture that most sites ignore this flag and instead use it to track clients anyway, I would believe it okay to eliminate this flag to disable fingerprinting by DNT.

0.7 Enable Popup

1. Upon loading, my JavaScript will attempt to `window.open` a new window. If this cannot be done (i.e. the newly opened object is `null`), then popup has been blocked.

2. `window.open` would need to be eliminated from native JavaScript calls to eliminate this feature. However, it seems that websites wouldn't lose too much without this feature. For one, the method to track whether popup is enabled is pretty intrusive: It is basically impossible to open up a new window without the client noticing. For another, a lot of modern browsers like Chrome and Firefox by default block popups unless configured by the client. Since people tend to leave settings as defaults, I don't think this is a key distinguishing feature for browser fingerprinting and wouldn't put too much efforts into eliminating it. (On a side note, if we are considering multi-site instead of single-site tracking, maybe distinguishing between the sites on which a client has blocked popups and those on which they have enabled popups is a decent tracking metric.)

0.8 Fonts

1. I made two `<span>` elements, left one in browser-default font and changed the font for the other. If the width of the two spans differ upon comparison, then the font of the second div must have been installed on the client's machine.

2. It is pretty difficult to prevent this type of tracking as the JavaScript is doing nothing extravagant other than changing fonts. A large set of fonts can serve a key distinguishing feature since it is very unlikely that two clients have exactly the same set of fonts installed. (Font differences may arise from different device model, OS, browser type and version, and maybe even software installed since some software comes with their unique fonts.) The same browser usability and user experience consideration from window size and user agent holds: Although font detection may reveal a lot of information with privacy complication, knowing what fonts a client has installed will help websites deliver better, more customized contents. It is in essence a convenience-privacy trade-off.

Problem 4

Cookies could expire in a month or stay permanent. However, a client may also delete their cookies whenever they want or disable cookies for all future visits. Therefore, cookies are a relatively unstable feature for tracking and websites could only rely on other feature mappings to associate the pre-change and post-change visits and reassign cookies to the client. **IP** is a relatively stable feature for tracking, assuming the client isn't connecting over an VPN and wouldn't move or go on travel in a month. **Window size** and `user agent` should be relatively stable features since it isn't very likely that a client will switch to a different device or a completely different browser in a month. **EnableCookies, EnablePopup** are relatively stable features because it is unlikely that a client will enable cookies and popups during their visit to a specific site once and then disable these features on another visit (and vice versa). I'd say **DNT** is a very stable feature since few people know about or care to set it, and their behavior probably won't change in a month (unless they take a computer security course :)). Lastly, **fonts** could potentially be unstable. Imaginably, the set of fonts would grow overtime as a client install new software without actively deleting unused fonts. To associate pre-change and post-change, visits, websites may observe whether the font set post-change is a superset of the one pre-change. In summary, the moral of the story seems to be that most of these tracking features are quite stable in a short time span (i.e. a month) but will become increasingly unstable overtime. (Given a year or two, stable ones like IP and user agent may change dramatically.) It is justifiable how these features are relatively stable, as the most ideal features for device fingerprinting should be unique, traceable, and stable.

Additional Comments

The most useful query I came up with to associate visits, using `JSON_ARRAYAGG` is:

```
SELECT page FROM (
    SELECT cookie, ip, window_width, window_height, user_agent, enable_cookie,
        do_not_track, enable_popup, fonts, JSON_ARRAYAGG(page) AS page FROM
        problem4
    GROUP BY cookie, ip, window_width, window_height, user_agent, enable_cookie,
        do_not_track, enable_popup, fonts
    ) as T;
```

Reference

Generate random string as cookie values: https:// gist .github.com/6174/6062387
Getting and setting cookies: https://www.w3schools.com/js/js_cookies.asp
Detecting popups: https://stackoverflow.com/questions/2914/how-can-i-detect-if-a-browser-is-blocking-a-popup
Getting the window size browser-independent: https://stackoverflow.com/a/28241682