


NWI-IMC074 Online Tracking and Privacy - Assignment 1

 Grade weight: **2.5/10** of the final grade

 Due: **6 March** 2024 (23:59, CET)

Step 1: Pick a website from the following list (online pharmacies and shops)

Websites	
<ul style="list-style-type: none">• apotheekehuid.nl• deonlinedrogist.nl• gezonderwinkelen.nl• koopjesdrogisterij.nl• medimart.nl• medischevakhandel.nl• merkala.nl• vitatheek.nl	<ul style="list-style-type: none">• coolblue.nl• bol.com• ah.nl• zalando.nl• jumbo.com• debijenkorf.nl• hm.com• douglas.nl

If you prefer, you can also use a pharmacy website or an online shop from your own country.

Step 2: Capture the HTTP traffic while accepting cookies (and personal data processing)

1. Create a new Chrome/Chromium profile
2. Open the DevTools/Network panel
3. Check "Preserve log" (that'll retain all requests made during a session)
4. Load the website you picked in Step 1; accept all cookies/data processing, dismiss other potential dialogs (permission to send notifications, location access, email signup etc.)
5. Scroll down until the bottom of the page
6. Click on a product to go to its (product) page. Multiple clicks are okay, if you have to go to a product page. Avoid external links; the product page should be under the same first-party domain as the homepage
7. Add the product to shopping cart and wait ~5 seconds (so the requests & responses are finalized)
8. Save all HTTP request/responses as HAR to a file using the following naming convention: example.com_accept.har. No www. or other prefixes; just domain_name_accept.har.

Step 3: Capture the HTTP traffic while rejecting cookies (and personal data processing)

Now, repeat steps 1-8 **starting again with a fresh profile**, but this time **reject all cookies and data processing in Step 4**. Feel free to make multiple clicks for rejection. Name the second HAR file as `domain_name_reject.har`.

Step 4: Analyze the HAR Data



Write an analysis script as a Jupyter Notebook (.ipynb) or as a standalone Python script (.py) that processes the captured HAR files and outputs two separate JSON files, each containing a serialized (Python) dictionary of results. The overall processing pipeline should look like the following:

❖ HAR -> analyze -> results dict -> save as JSON

The results dictionary serialized in each JSON should contain the following keys:

- `num_reqs`: Integer; number of requests (observed in the HAR file)
- `num_requests_w_cookies`: Integer; number of requests with a non-empty Cookie header
- `num_responses_w_cookies`: Integer; number of responses with a non-empty Set-Cookie header
- `third_party_domains`: List[String]; list of distinct third-party domains (eTLD+1)
- `tracker_cookie_domains`: List[String]; list of distinct domains that set a cookie that can be used for cross-site tracking. Only consider cookies set by HTTP response headers. You don't need to investigate whether a cookie is really used for tracking or not. Having necessary cookie attributes (see, lectures) for cross-site tracking is enough. Assume a minimum (cookie) lifespan of 60 days (based on Max-Age or Expires attributes)
- `third_party_entities`: List[String]; list of distinct entity (i.e. company/organization) names that own the domain names of the request urls (based on [DuckDuckGo's domain -> entity map](#))
- `requests`: a list of dictionaries, where each dictionary contains the following request/response details:
 - `url_first_128_char`: String; the first 128 characters of the request URL; e.g. `https://example.com/pixel.gif`
 - `url_domain`: String; e.g. `example.com`
 - `is_third_party`: Boolean; does the request have the same eTLD+1 as the website
 - `set_http_cookies`: Boolean; whether the response has a non-empty Set-Cookie header
 - `entity_name`: String; Name of the entity (or 'unknown') that owns the request url domain. (Again, using [DuckDuckGo's domain -> entity map](#))

Tips:

- The req_resp_pairs list will contain one dictionary for each request-response pair
- To save the HAR file: On DevTools network panel: 1) Right-click -> Copy-> Copy all as HAR -> Paste to an empty file. If that doesn't work (e.g. encoding issues when parsing the JSON) try 2) Right-click -> Save all as HAR with content.
- Unless specified, “domain” means eTLD+1 (aka., *registrable domain or payable domain*)
- Comment your code when what you do is not obvious
- DRY: Don't Repeat Yourself. Break your code into reusable small functions
- Avoid deep code indentations
- Use meaningful variable and function names
 - a.  good: request_domain, response_headers, get_entity_by_request_url
 - b.  not good: foo, bar, tmp, do_stuff

Practicalities

- Upload a zip file containing the files listed below (a-g). Name the zip file after your student number; e.g. s012345.zip. File names inside the zip archive should look like this:
 - a. example.com_accept.har
 - b. example.com_accept.json
 - c. example.com_reject.har
 - d. example.com_reject.json
 - e. s012345.ipynb OR s012345.py (analysis script)
 - f. requirements.txt: Python packages required to run your script, if any
 - g. domain_map.json ([link](#))
- You are free to use publicly available Python packages (e.g. to parse dates).
- Your code should **not** make any calls to online APIs. It should be able to work offline.
- You can print log messages from your code (you don't have to)
- Your code should work with Python 3
- Your code should be able to run without any command line parameters
 - a. Hard-code the HAR filenames in your code, assume they are in the same folder as the analysis script/notebook
 - b. Standalone scripts: Running “python s012345.py” once should re-generate both JSON outputs with the same content as the submitted JSONs (i.e., the results should be reproducible)
 - c. Jupyter Notebooks: Running the notebooks cells should be possible without any intervention and it should re-generate the exact JSON outputs

Help:

- You can ask your questions on Brightspace, if anything is unclear:
<https://brightspace.ru.nl/d2l/le/427013/discussions/topics/109635/View>

 **Good luck!** 