MASTER IN COMPUTER SCIENCE

unine UNIVERSITÉ DE NEUCHÂTEL

Assignment 8 - HTTP proxy (part 2)

Security

Sébastien Vaucher
sebastien.vaucher@unine.ch

5 December 2017

1 Assignment instructions

In this assignment, you will add caching capabilities to the HTTP proxy server that you developed in the previous assignment. Caching means that your proxy will locally keep a copy of any page suitable for caching. Thanks to this mechanism, bandwidth usage can be greatly reduced for static content.

Your cache implementation will adhere to the following requirements:

1. It may only cache content if allowed by the official HTTP/1.1 specification [1];
2. The cache must be persisted to disk, so that it survives a restart of the proxy;
   a) Alternatively, an external key-value store can be used as persistent storage (e.g. Redis, Memcached, ...);
3. The proxy will do positive and negative[1] caching;
4. You can assume that once a page is cached, it stays cached forever. You can therefore ignore problems like freshness of data, re-validation of queries, or cache expiration;
   a) You are obviously still allowed to implement these features; bonus points shall be awarded for their implementation.

[1] With negative caching, the cache keeps a list of unavailable pages in cache. When a query for one of these pages comes in, it immediately replies with empty data.

2 Hints

- A simple approach is to store each received web-page in its own file;
- Using hashed values as keys for cached content may simplify your work, as you do not need to handle special characters anymore;
- The ultimate source of truth for this assignment is RFC7234 [1];
- You obviously do not need to implement the whole specification! You should only implement the parts that you feel are relevant. Nonetheless, you must not go against the specification;
- The most important headers to parse to determine if a page can be stored in cache are `Cache-Control` and `Expires`.

3 Hand-in

The time allotted for this assignment is 1 week. The deadline is on 2017-12-13T13:59:59 local time. Late submissions are not accepted.

- To be submitted to Ilias[2]:
  - Source code of **your** assignment
  - Readme file briefly mentioning how to compile and run your program, which dependencies it requires, etc.
  - All the files have to be packed in an archive in a standard format[3], named following this exact pattern (in lowercase letters only):
    `security17-as<assignment number>-<your family name>.<extension>`.
    For example, if your name were to be *Homer J. Simpson*, you would use the following filename for this assignment: `security17-as8-simpson.tar.gz`
  - Please use the "Upload File" button when handing-in your assignment in Ilias. Do **not** use "Upload Multiple Files as Zip-Archive".
- You have to present a demonstration of the program in class (to the TA).
  - It is **mandatory** for each student to demonstrate his or her submission!
  - The sooner you present your assignment, the better (even before the deadline).

Your grade will depend on both the presentation and the code.

[2] Or sent by e-mail for external students

[3] `.tar`, `.tar.gz`, `.tar.bz2`, `.tar.xz`, `.zip`

4 Notes

You can use your favorite programming language for the assignments of this course, so long as it is a programming language readily available on the GNU/Linux operating system[4].

Should you have additional questions, please direct them to the TA at sebastien.vaucher@unine.ch.

References

[1] R. Fielding, M. Nottingham, and J. Reschke, "Hypertext Transfer Protocol (HTTP/1.1): Caching," RFC Editor, RFC 7234, Jun. 2014. [Online]. Available: `https://tools.ietf.org/html/rfc7234`.

[4] You can use any of the languages in the following list. If you want to use another language, please check with the TA first. List in alphabetical order: Bash, C, C++, Go, Java, Kotlin, Perl, PHP, Python, Ruby, Rust, Scala