CS112 Final Project

Goals:

1. Building a complex networked system (substantially larger in scope than the assignments).
2. Applying concepts learnt in class (and beyond, on your own) in your design
3. Learning to work effectively in groups and achieving something that you won't be able to accomplish individually.

Logistics

1. Projects should be done in teams of 2 (or 3 with instructor's approval). We discourage working individually, but you are allowed to do so if it is unavoidable. Within each project, each member should own a substantial piece of functionality/code.
2. You can work in any language of your choice. We encourage you to continue working in C -- so you can build more expertise in C system level programming, a useful skill for your future careers.
3. The important deliverables include the artifact submission (code+report), final demo/presentation, and a short project report.
4. You should use GitHub (or similar) to manage your artifact (e.g., code, dataset etc.) and also share it with us. Additionally, all groups are expected to ensure their artifacts are easy to set up (e.g., by creating a Docker container) and the results are easy to reproduce. Further details on artifact submission requirements will be provided later.

Project Ideas

1. We are providing high-level guidelines for a suggested project on building an HTTPS proxy powered by Large Language Models (LLMs). This project is intended to give you an understanding of the complexity expected.
2. You can also come up with your own idea, something that's more exciting for you -- but your project should be challenging enough to match the suggested project.

Deliverables

All deadlines are 5PM EST unless otherwise specified.

1. (11/22) Checkpoint 1. Expectation: baseline requirements (detailed later). You will be required to give a short (15 minute) demo to the instructor/TAs.
2. (12/06) Checkpoint 2. Expectation: baseline complete + most of the advanced features. You will be required to give a short (15 minute) demo to the instructor/TAs.
3. (12/10) Checkpoint 3. Expectation: complete implementation as well as extensive evaluation. You will be required to submit your artifact and also prepare a short artifact report (max 1 page) detailing how to run your system/reproduce the main results.
4. (12/12) Final Project Due. Final Project Demos/Presentations and Report.

HTTPS Proxy powered by Large Language Models (LLMs).
We are providing high level guidelines for the design of an LLM powered HTTPS proxy.
The main idea is that the proxy should work with HTTPS and leverage LLMs (e.g., GPT4) to enhance the data (e.g., website) sent to requesting clients.

The main steps include: forwarding the client request to the server, decrypting the data sent by the responding server, using an LLM (we will provide API-based access) to enhance it in some way, re-encrypting and sending it back to the client. The design of LLM based enhancement features are up to you.

Note: You are encouraged to use these as guidelines only (rather than specifications) and come up with your design.

Baseline Features: Must be completed by Checkpoint-1

1. *Basic support.* GET method, caching (similar to a1), ability to handle multiple clients (using select/threading etc.), error handling (similar to a2). Ideally you should be building on your implementations for a1 and a2.
2. *HTTPS support.* This involves handling the CONNECT method of HTTP and implementing SSL interception. Your proxy should be trusted by the clients (e.g., browser[1]) so it can handle SSL connections while being able to look at the content. This will require your proxy to decrypt/encrypt content that it is getting from the server.
3. *Basic Performance Testing and Optimizations*. Your proxy's performance should be comparable to the baseline (without using a proxy). Note that this is different from a1 where performance wasn't a consideration.

Advanced LLM related Functionality: Most of this should be completed by Checkpoint-2
LLMs can be used to enhance/transform server responses. Some potential directions include:

1. *Advanced Performance Optimizations*: Can your proxy actually improve web download times compared to the baseline? LLMs could reduce the size of server responses (e.g., by summarizing), help you understand which objects are more important than others, help in designing content prefetching strategies, and so on. The goal would be to show faster page-load times/reduced bandwidth usage
2. *Accessibility Features:* Can your proxy make web browsing more (or less) accessible? LLMs can be used to enhance readability (facts, summaries), make text interactive (e.g., pop-up that shows additional context/definition etc.), add/remove images, filter content (useful for forums requiring real-time moderation), personalize responses
3. *Offline Proxy Cache:* Your proxy doesn't have access to the internet anymore (only to a small language model); can it provide a rich browsing experience using the cached content? This will require building a basic search engine on top of the cached content

You are also encouraged to come up with your own features (feel free to share those on Piazza). Also, check back here for possibly additional feature suggestions.

---

1. You will need to install a root certificate in your browser. The proxy will use the certificate to establish a secure connection with the client

Extensive Benchmarking and Stress Testing: Must be completed by Checkpoint-3
Once your proxy is fully implemented, we expect you to extensively benchmark it: testing it against the top 100 Alexa websites; deploying it on various cloud locations; understanding where benefits come and where it hurts performance, etc.

For some LLM-based features, the evaluation will be more nuanced. It will be useful to have a plan for evaluating those. Please discuss your strategy with the TA in advance.

You should plan to spend at least a week-10 days on this. We expect that the final report will contain your results and experiences from this phase. For groups of three students, the bar for this checkpoint is higher.

LLM Access:
We will provide an API to access LLMs and make requests. The TA will conduct a tutorial to go over the API usage.

API key: To use the API, you will need a key. Please reach out to the course staff to be provided with one. There is a usage quota associated with each key (max rate: 1 req/sec, max of 128 requests/day, max 5 concurrent requests). **DO NOT** share the API key with members outside your group.