Exercises for Module 1

The exercises for the first module are about LLM basics, prompting, LangChain and LangGraph. Supporting code can be found at https://github.com/henningth/Generative-AI-in-Cybersecurity

Course setup

Exercise 0:

(a): Make sure you've got a working installation of Python and an IDE. I use Visual Studio Code, you are free to choose other editors.

(b): Register an account at OpenAI (note that this is not a ChatGPT subscription): https://platform.openai.com. To use the OpenAI API, you need to pay a certain amount. For this course, $10 is sufficient.

(c): Register a free account at GroqCloud: https://console.groq.com. Here you can use Open-source models in the cloud (the free tier has limits on tokens etc.)

LLM basics

In these exercises, we will be examining various LLMs in terms of their capabilities. This includes how well they summarize text and code, classification capabilities and reasoning, among other things. We will be using both closed-source models (via OpenAI) and open-source (via Groq). Begin by downloading the Python script 01_llm_basics.py from the Github page.

In your Python editor, create a file called .env. Generate an API-key from OpenAI and Groq, and insert them into the file like shown below:

OPENAI_API_KEY="sk-..."

GROQ_API_KEY="gsk..."

In these exercises, experiment with various model types and sizes (i.e. number of parameters). Also experiment with various values of the temperature parameter.

Exercise 1: Study how well the model retrieves information and the accuracy of the information. Suggested prompts:

"Who is the current CEO of Microsoft?"

"What are the main symptoms of diabetes?"

"When was the GDPR enacted, and what is its primary purpose?"

"Summarize the key findings of the SolarWinds cyberattack."

Exercise 2: In this exercise, you will write and test prompts for various tasks. The purpose of this exercise is to test how well the various LLMs process these prompts, and the quality, relevance and format of the output. Formulate and test prompts for these tasks:

Generate email text in various tones

Translate code from one language to another

Generate code in a given language

Exercise 3: Zero-Shot vs. Few-Shot Prompting for Spam Classification

In this exercise, you will investigate how different prompting strategies—zero-shot and few-shot—affect the performance of Large Language Models (LLMs) on a common cybersecurity task: classifying emails as spam or not spam.

Part A: Zero-Shot Prompting

1. Select a small set of email texts from the set below, mixing obvious spam and legitimate emails.
2. For each message, prompt the LLM without any examples, using a simple instruction like:
   "Classify the following email as spam or not spam: [email text]"
3. Record the model's response and note any uncertainty or inconsistency in the classification.

Part B: Few-Shot Prompting

1. Choose 2–3 example email classifications (e.g., one spam and one legitimate) and include them in the prompt before the test email. For example:

"Example 1:
Email: 'You've won a $1000 gift card! Click here to claim.'
Classification: spam

Example 2:
Email: 'Meeting moved to 2 PM. See you there.'
Classification: not spam

Now classify this email:
[new email text]"

2. Repeat the classification for the same set of emails used in Part A.
3. Compare the model's output with the zero-shot results.

The sample emails that you will use are below:

Legitimate (non-spam) emails:

1. Subject: Lunch plans?

Body:

Hey! Are you free to grab lunch on Thursday? Thinking about trying the new Thai place.

— Alex

2. Subject: Your GitHub pull request was merged

Body:

Great job! Your changes have been merged into the main branch. Thanks for the contribution.

3. Subject: Conference registration confirmation

Body:

Thank you for registering for the 2025 Cybersecurity Summit. This email confirms your participation.

Spam emails:

1. Subject: Claim your $1000 gift card now!

Body:

You've been selected to receive a $1000 Amazon gift card. Click here to claim your reward before it expires!

2. Subject: Final notice: Update your payment info

Body:

Your subscription will be cancelled unless you update your billing information now using this link.

3. Subject: You've won a free vacation!

Body:

Congratulations! You've won an all-expenses-paid trip to the Bahamas. Call now to book your spot.

Exercise 4: Code Summarization with LLMs

In this exercise, you will explore how well different Large Language Models (LLMs) can summarize source code.

Tasks:

- Choose a code snippet or script from the Software Security or Offensive IT Security course materials.
- Use at least two different LLMs to generate summaries of the selected code.
- Compare the quality, accuracy, and clarity of each model's summary.
- Discuss which model is more helpful for understanding the code's purpose and functionality.

Exercise 5: Summarizing CVE Reports and Exploit Code with LLMs

Building on Exercise 4, you will now test the ability of LLMs to summarize real-world security vulnerabilities and associated proof-of-concept (PoC) exploit code.

Instructions:

For each of the CVEs listed below, collect information from the following three sources:

- National Vulnerability Database (NVD): https://nvd.nist.gov/
- Security Blog Posts: For example, from Cloudflare, CrowdStrike, Rapid7, etc.
- Proof-of-Concept Exploit Code: Locate PoCs via NVD references or on GitHub.

Tasks:

Use LLMs to generate concise, technical summaries for each CVE based on the above sources.

Evaluate how well the LLMs:

- Explain the root cause and exploit mechanism
- Summarize the PoC code (without running it)
- Identify potential mitigations
- Compare and critique the summaries from different LLMs.

Target CVEs:

(a) CVE-2021-44228 – Log4Shell (Apache Log4j Remote Code Execution)

(b) CVE-2017-5638 – Apache Struts2 OGNL Injection

(c) CVE-2019-19781 – Citrix ADC Directory Traversal