10 best practices for securely developing with AI

AI-assisted applications

01 Prompt injection

Prompt injection (similar to SQL injection) is the manipulation of an LLM by providing inputs designed to trick the LLM into performing tasks it shouldn't be allowed to do.

- Educate your teams on prompt injection, and consider gamifying the learning with a tool like Gandalf from Lakera.
- Use the principles of least privilege between your LLM and your data/functionality.
- Limit the sensitive data your LLM has access to.
- Treat your LLM like user data and sanitize actions and responses.
- Use function calling where possible to avoid unstructured data which might change the context for the LLM/desired behavior
- Be familiar with indirect prompt injection as well, in which data sources are poisoned
- Avoid user supplied prompts directly, unless necessary, is the best form of defense.

02 Restrict data access for your LLM

- Treat your LLM like your user data, and be careful of direct access between your LLM and data.
- Adhere to the rules of least privilege.
- Don't give your LLM more data than it needs to do its job.

Add input and output guard checks around your LLM interactions to sanitize input from users and output from LLM.

03 Get to know the OWASP Top 10 for LLMs

OWASP has produced a top 10 list for LLMs based on the views of nearly 500 experts, including Snyk. Check out our OWASP Top 10 Considerations for Addressing LLM Risks cheat sheet.

AI-assisted development

04 Keep a human in the loop

For developers: Use caution around AI-generated code: validate during code review and integrate code security tooling into the IDE (like Snyk) to test first-party code and third-party/OSS libraries.

For apps: This is particularly important when using autonomous agents available through Langchain etc., which give AI direct exec access.

- Use caution around allowing your LLM to execute functions/system calls.
- Use caution around allowing your LLM to manipulate/change important/sensitive data.

05 Secure your vulnerabilities

- Treat generated code, like inexperienced developer code: validate, test, and correct in code reviews primarily in the IDE.
- Use tools like Snyk Code to automate the testing of your AI-generated code.
- Test and fix first-party code in the IDE (where code is generated).
- Consider always manually verifying open source libraries recommended by AI. And use tools like Snyk Open Source and Snyk Advisor to test AI-suggested open source libraries.

06 Don't provide IP/private info to public GPT engines

- Don't share anything you wouldn't want to be publicly known. Assume anything you put into a public AI engine will be used for future training.
- Investigate enterprise-ready versions of your AI tooling.
- Educate your teams on acceptable policy when using GPT tools.

AI models

07 Use hybrid AI models where you can

- Use the right tool for the job. Consider all forms of AI, including symbolic AI, as is used in Snyk Code, to represent knowledge using symbols, rules, and logic to perform tasks.
- Symbolic AI's main use case is to build intelligence into very small domains using **limited** data sets. This is great when data has to follow specific formatting and rules.
- LLMs are best at creating broad answers with general purpose problems where absolute accuracy is less important. Symbolic AI is great at fine-tuning and validating generic answers.

08 Use good training data

- Fine-tune your LLM model (expensive, slow)
- Use different in-context learning techniques (fast, fairly accurate, easy, cheap)
- Validate/verify data sources.
- Sandbox data sources so external data can't be added easily.
- Perform attestation/signing of data.

09 Beware of hallucinations and misleading data

- LLMs will hallucinate and confidently tell you they are right. You always need to validate their output.
- Treat your LLM like your user data and validate and sanitize its output.
- Don't allow your LLM to be able to execute dangerous functions without validation.
- Consider human interaction as needed to avoid data affecting critical systems/sensitive data

10 Keep track of your AI supply chain

- List the data source dependencies you're using for the training/tuning of your LLM.
- Use attestation/signing techniques to validate the data you use.
- If AI recommends usage of tools, or SDKS, responsibly evaluate and validate everything manually. Attackers will spoof SDKs and tools that AIs are likely to recommend.

snyk

Mitigate AI-generated code security risks for free with Snyk.

Learn more ›