LLM APPLICATIONS SECURITY CANVAS

v2.3

Protect against jailbreaks and prompt injections

Untrusted input sources

RATE LIMITS

Inspect / Sanitize

- Contextual, use-case specific **input validation** (length, format, language, character set, ...)
- Block common **Jailbreak /Prompt Injection** patterns
- Moderate inputs for general **harm categories** such as hate speech, violence, self-harm, ...
- Apply **topical guardrails / semantic routing** to ensure user queries match intended scope / use-case and are routed accordingly

Instruction / Data Separation

Use **Spotlighting** / prompt engineering techniques to help the LLM distinguish instructions from data

- **Border strings**
- **Data-marking**
- **Multi-turn dialogue**

Implementation

Classic block/allow-lists

- Exact text matching and regex

Machine learning methods

- **Semantic search** using embeddings
- **Classifiers**, such as BERT
- **LLM-as-judge**

Prompt

LARGE LANGUAGE MODEL (LLM)

Output

Inspect / Sanitize

- Contextual, use-case specific **output validation**
- Allow only **expected content types/formats** (strictly validate JSON/XML field values)
- **Block** potentially **dangerous content** like URLs, JavaScript, or markdown images, unless explicitly validated through an allow-list.
- Ensure **safe rendering** in web apps (contextual output encoding, Content Security Policy, ...)
- Moderate outputs for **harm categories** like hate speech, violence, self-harm, ...
- Real-time **hallucination** and **offtopic** checks for responses

THE RULES FOR USING LLM

Rule #1 Treat the LLM as untrusted
Rule #2 Validate LLM outputs
Rule #3 Validate LLM inputs
Rule #4 Implement adaptive content moderation and suspend offenders

Tool / Function Calls

Access Control

- Least privilege
- Downstream checks
- Sandboxing

Safe APIs

- Limit function / scope of allowed operations
- Ensure APIs/functions are not vulnerable themselves (OWASP Top 10)

Human in the loop All sensitive actions are approved by humans

REVERSEC