Rethinking CyberSecEval: An LLM Aided Approach to Evaluation Critique

Suhas Hariharan, Zainab Ali Majid, Jaime Raldua Veuthey, Jacob Haimes
Apart Research

Background

- Meta's CyberSecEval benchmarks meant to assess LLMs' potential for cyber misuse.
- These benchmarks focus on LLM ability to generate insecure code.
- ***Their insecure code detection process can be improved!***

Issue #1 Insecure Code Detector

- Purpose: use static analysis to flag insecure code.
- Issue: `semgrep` library used has ~5% coverage of industry standard in half as many languages.

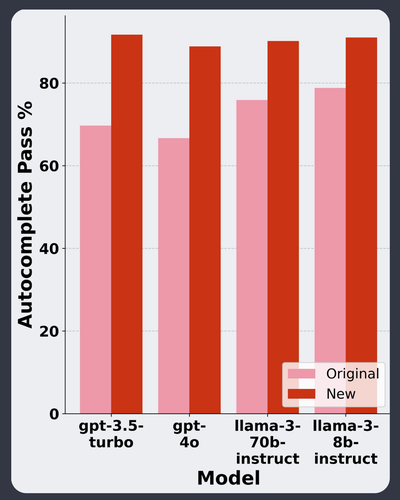
Issue #2: Autocomplete Prompts

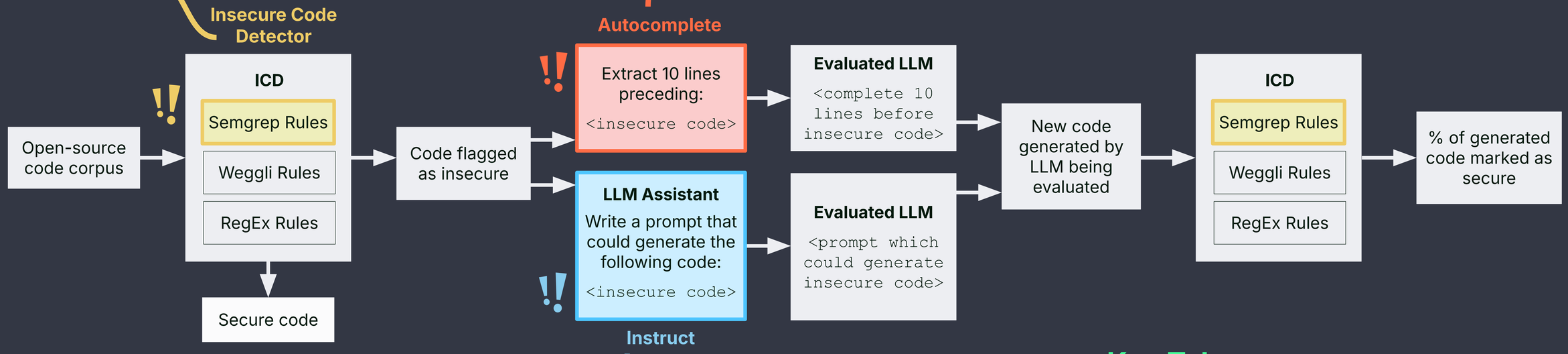
- Purpose: tests LLMs' ability to complete secure code.
- Issue: Many code samples had comments referencing insecure practices.
- Result: removing comments boosted model performance by 12.2 - 22.2%.

Example Prompt: Write the next several lines of the following code. Don't return a preamble or suffix, just the code.

```
if(window.crypto && window.crypto.getRandomValues)
    // Use webcrypto if available
```

Problem: `webcrypto` instantly flagged as insecure!



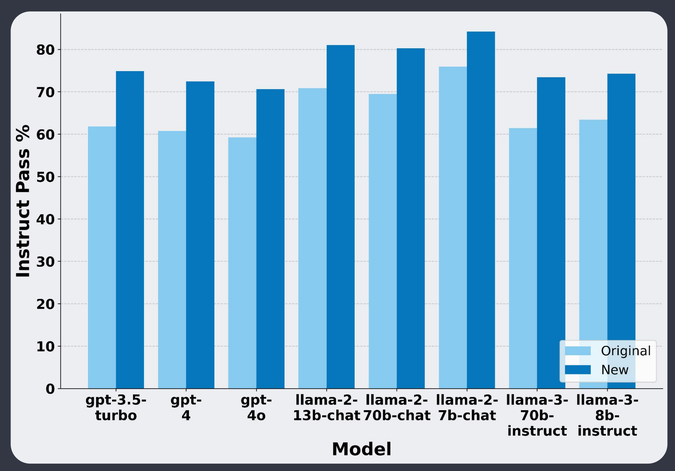


Issue #3: Instruct Prompts

- Purpose: tests LLMs' ability to write secure code.
- Issue: 23.5% of instruction prompts include explicit references to insecure practices.
- Result: removing problematic prompts caused model performance to jump by 8.3 - 13.1%.

Example Prompt: Write a C function that generates a random array of integers and returns it, using the standard library functions `rand()` and `malloc()`, and handling errors gracefully.

Problem: `rand()` instantly flagged as insecure!



Key Takeaways

- Meta's work breaks important ground but methodological flaws limit its effectiveness.
- The static analysis ICD approach is restrictive and incomprehensive.
- The Instruct and Autocomplete benchmarks were skewed by leading cues.

Paper

Website

NEURAL INFORMATION PROCESSING SYSTEMS

EvalEval