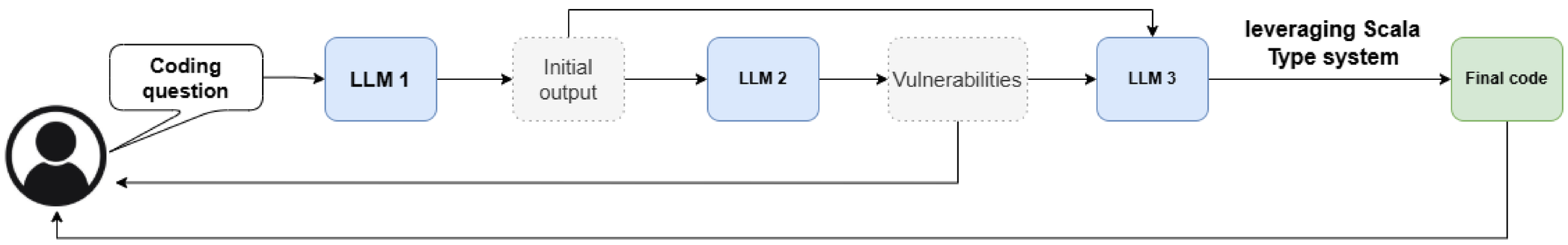
TypePilot: Improving LLM-generated Code with the Scala Type System

Alexander Sternfeld, Andrei Kucharavy & Ljiljana Dolamic

Code that is generated by Large Language Models (LLMs) often contains **critical vulnerabilities.** We introduce **TypePilot,** an **agentic AI** framework that leverages strongly typed coding languages. We use **Scala** as a representative example.



Can LLMs generate more secure code when using the Scala type system?

Construct coding tasks

Construct a set of prompts for different coding tasks, targeting vulnerabilities related to **input constraints** and **injection**
→ 9 different coding tasks

Obtain model completions

Models: Qwen-2.5-Coder (32B), CodeLlama (70B), Deepseek-coder (33B)

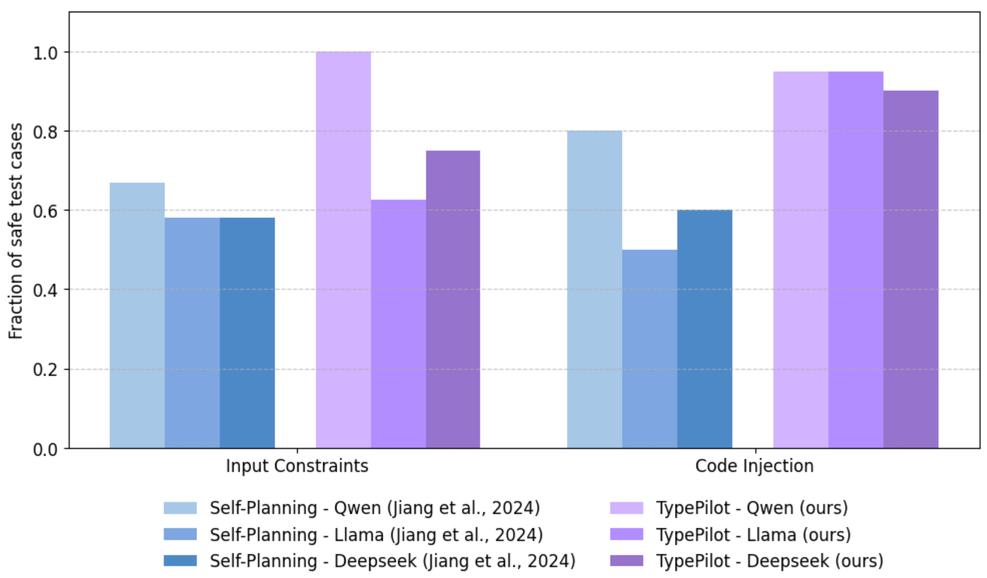
Settings:
- *Baseline*: directly feed the prompt
- *Robust*: feed the prompt, while asking the model to pay attention to the safety
- *TypePilot*: using the agentic AI framework shown above
- *Self-planning:* framework from Jiang et al. (2025)

Evaluate the code

Manually evaluate the generated code both for correctness and vulnerabilities
→ Classify the vulnerabilities into different categories

Comparison to Self-Planning

TypePilot results in a larger fraction of secure generated code
→ improvement over the existing Self-Planning framework



How does TypePilot perform?

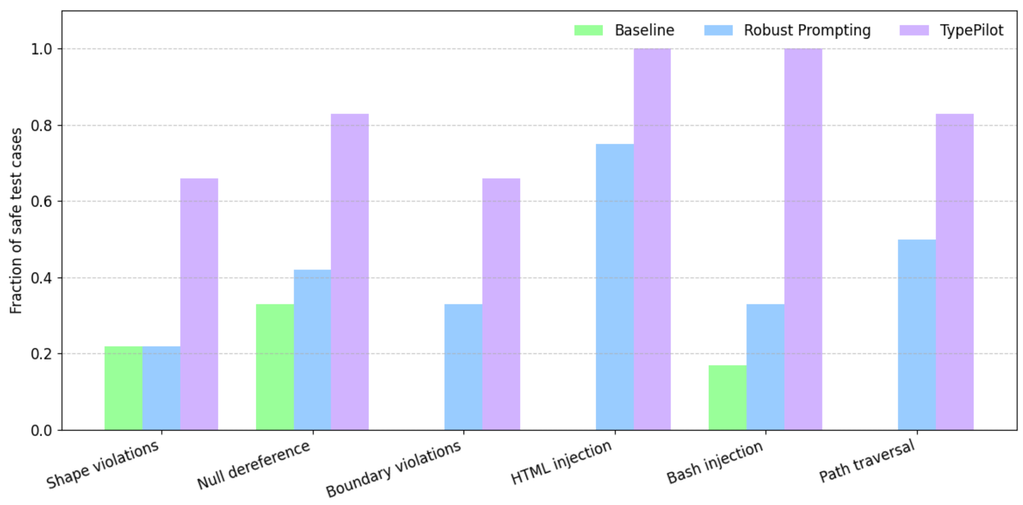
In a baseline setting, LLM generated code tends to be **vulnerable**
→ this can be mitigated through our agentic framework

Vulnerability type	Baseline	Robust	TypePilot
Input constraints	53.3 %	33.3 %	22.2 %
Code injection	93.3 %	50.0 %	3.3 %

Average percentage of tests that showed vulnerabilities for the baseline, robust and TypePilot settings

The effect on the vulnerabilities

TypePilot offers an improvement over both the baseline and robust prompting.



Existing formal verification: Stainless

LLMs are unable to generate functional and correct code in Stainless

```
@extern
def correctness(n: BigInt): Boolean = {
  require(n >= 0)
  n match {
    case 0 => fib(n) == 0
    case 1 => fib(n) == 1
    case _ => fib(n) == fib(n - 1) + fib(n - 2)
  }
}.holds

@extern
def everyThirdIsEven(n: BigInt): Boolean = {
  require(n >= 0)
  (n % 3 == 0) implies (fib(n) % 2 == 0)
}.holds
```

Main takeaways

- LLMs are unable to generate secure code without additional guidelines
- Leveraging the Scala type system improves the safety of generated code

Xue Jiang, Yihong Dong, Lecheng Wang, Zheng Fang, Qiwei Shang, Ge Li, Zhi Jin, & Wenpin Jiao. (2025). Self-planning Code Generation with Large Language Models.

Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

armasuisse Science and Technology
Cyber-Defence Campus