経験的アプローチによる自動プログラム修復の提案
An Empirical Based Proposal for Automatic Program Repair

髙橋 モハマドシャールク・ネットワーク分科会・情報セキュリティ大学院大学

Software bugs are defects or glitches present in computer programs or systems. They pose risks to software providers, prompting various attempts to eliminate them. Automatic Program Repair (APR) is one such approach. While APR is an active field in both development and research, it faces challenges in the number of executable patches generated and its ability to address security bugs. This study attempts to overcome these challenges by leveraging experience in patch generation using Large Language Models (LLMs).

背景

- 自動プログラム修復（以下、「APR」と書く）とはソフトウェアバグを自動で修復することである
- 近年APRの開発および研究が盛んにされているが開発面および研究面ともに課題を抱えている

開発面の課題：セキュリティバグの修復能力

3月20日（米国時間）より、GitHub Advanced Securityをご利用のすべてのお客様を対象に、**Code Scanningの自動修正機能**がパブリックベータ版として提供されます。GitHub CopilotとCodeQLを利用したCode Scanningの自動修正機能は、JavaScript、TypeScript、Java、Pythonのアラートタイプを90%以上カバーし、検出された脆弱性の ⅔ 以上のほとんど、あるいはまったく編集することなく修正できるコードを提案するものです。

研究面の課題：セキュリティバグの修復能力、コンパイルに通るパッチの生成率

脆弱性の種類	信頼できるパッチの生成率	コンパイル成功率
CWE-787(バッファーオーバーフロー)	2.2%	約46%
CWE-89（SQLインジェクション）	29.6%	約98%

目的

両研究で共通する「セキュリティバグの修復能力」の向上

開発、研究の両観点から見える課題
セキュリティバグを念頭に置いていない
生成したパッチのほとんどが実行可能なパッチとなっていない（コンパイルができない）
適切なパッチを生成するのに時間がかかる

提案手法

過去に生成したパッチに関する情報（つまり経験）を活用したAPR

- パッチ検証の結果等の情報のナレッジベースを作成
- ナレッジベースを参照してパッチを生成する ＝＞ RAGを活用

評価結果

APR手法	信頼できるパッチ(1)	信頼できるパッチ(2)	信頼できるパッチ(3)
ARJA	5	-	-
Tbar	5	-	-
Codex	6.2	-	-
InCoder	3	-	-
RewardR	2	-	-
本研究	0	0	-

考察

- コンパイルに通らないプログラムを修正するのは非常に困難
- 入力長が大きい場合のAPRでは入力プログラムとLLMで想定するバージョン不一致が修復能力に影響を及ぼす

今後

- 修復能力の向上
- セキュリティバグ以外の修復能力の検証
- DevSecOpsへの統合