ICML 2025
International Conference On Machine Learning

RepoAudit: An Autonomous LLM-Agent for Repository-Level Code Auditing

Jinyao Guo*, Chengpeng Wang*,

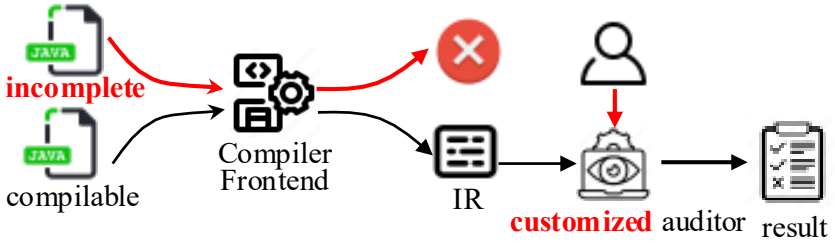
Xiangzhe Xu, Zian Su, Xiangyu Zhang

(* Equal Contribution)

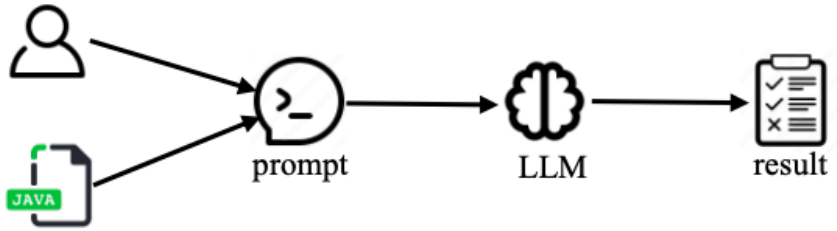
PURDUE UNIVERSITY

Code Auditing in LLM Era

- Writing coding is easy. High **code quality** is what matters.
- Code auditing for code quality improvement



Traditional code auditor



LLM-driven code auditor

Analyze compiler IR
- Rely on build
- Difficult to customize

Audit via prompting
- Build-free
- Easy to customize

LLMs are NOT Silver Bullets

Challenge: Non-linear contexts underlying programs exacerbate **LLM hallucinations**, potentially causing many **FPs/FNs**

- Non-linear contexts: Control-flow graph, data-flow graph, call graph, etc

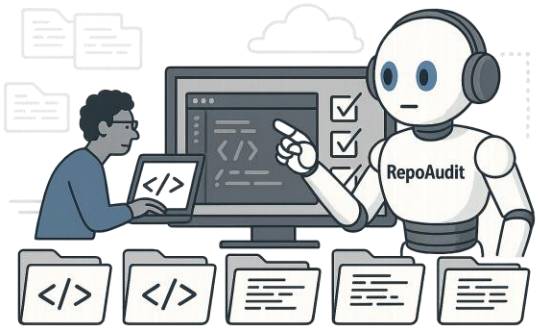
Preliminary Study

- Select 100 buggy functions before and after fixes
- Apply LLMs for bug detection
- Precision: **~30%**

Even worse in repo-level code auditing

- Even **lower precision and recall** due to **non-linearity of call graph**
- Potential **huge overhead** in scanning the repository

Auditing Code Repository as Human



Memory

- A single function is **more linear** than the whole repository
- Memorize **data flows** in **single functions**

Planning

- Start from functions containing sources (i.e., faulty values)
- Search for sinks (i.e., dangerous operands) by exploring callers/callees **on demand** for **better scalability**

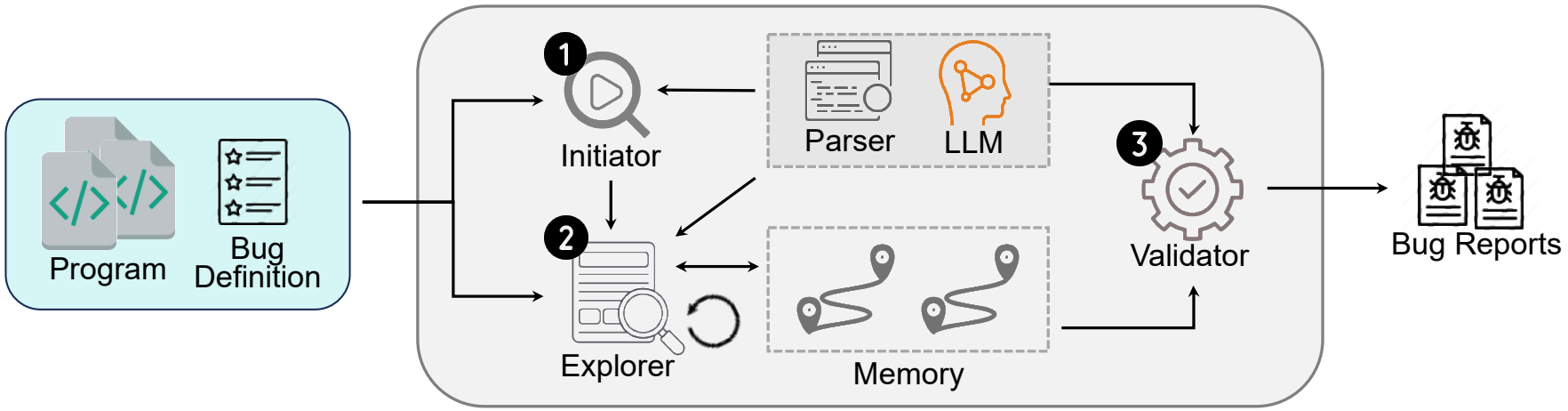
Tool using

- Value retrieval: Sources, sinks
- Function retrieval: Callers/callees

Workflow of RepoAudit

Design an agent by mimicking the process of manual code auditing

- **Divide-and-conquer**: Analyze code repository function-by-function
- Explore code repository in a **demand-driven** fashion
- Interpret single functions and validate data-flow path with **neuro-symbolic analysis**



Empirical Results with Claude-3.5-Sonnet

- **High precision:** **78.43%**
- **High recall:** Detect all the existing bugs
- **High efficiency:** **0.44 hours** and **$2.54** per project on average

Bug Type	ID	TP		FP	Feature		# Prompts	# Tokens		Financial ($)	Time (s)
		Old	New		# Intra	# Inter		Input	Output		
NPD	N1	1	(3,3)	2	0	4	145	709,919	55,863	2.97	2026.13
	N2	7	(1,0)	0	4	4	17	97,717	8,518	0.42	283.84
	N3	1	(1,0)	3	1	1	109	599,674	52,936	2.59	1747.90
	N4	1	(0,0)	1	0	1	29	126,852	13,654	0.59	435.09
	N5	1	(5,4)	1	0	6	63	420,710	31,375	1.73	1059.57
MLK	M1	1	(2,1)	1	2	1	205	1,132,763	85,279	4.68	2,917.91
	M2	1	(6,6)	2	4	3	146	845,148	71,243	3.60	2282.31
	M3	1	(0,0)	0	1	0	2	10,481	1,019	0.05	34.34
	M4	1	(0,0)	0	1	0	1	5691	619	0.03	17.94
	M5	1	(0,0)	0	1	0	35	181,348	20,779	0.86	599.92
UAF	U1	1	(0,0)	0	1	0	36	179,939	17,547	0.80	582.23
	U2	1	(0,0)	0	1	0	2	8900	869	0.04	31.95
	U3	1	(0,0)	0	1	0	48	317,713	23,067	1.30	791.98
	U4	1	(0,0)	0	1	0	10	48,087	5,883	0.23	185.22
	U5	1	(1,0)	1	1	1	662	4,534,444	303,645	18.15	10,661.98
Average							**100.67**	**614,625.73**	**46,153.07**	**2.54**	**1,577.22**

Comparison with Different Auditing Tools

Feature	RepoAudit	GitHub CodeQL	Meta Infer	Amazon CodeGuru	Cursor
Method	Neuro-symbolic	Symbolic	Symbolic	Neuro-symbolic	Neural
Detection Ability	**High**	Medium	Low	Medium	Medium
Accuracy	**High**	Medium	High	Low	Low
Multi-lingual	**High**	Partial	Low	Partial	High
Build-free	**Yes**	Partial	No	Yes	Yes

Real-world Impact

- **185 zero-day vulnerabilities** in high-profile open-source projects
- **95 bugs confirmed** and **79 bugs fixed**
- **Seven zero-days** in DARAPA AIxCC Ngnix Challenge
- Invited talks in RSAC 2025 and GitHub CodeQL team

31	challenge-004-nginx	C	NPD
32	challenge-004-nginx	C	NPD
33	challenge-004-nginx	C	NPD
34	challenge-004-nginx	C	Memory Leak
35	challenge-004-nginx	C	Memory Leak
36	challenge-004-nginx	C	Memory Leak
37	challenge-004-nginx	C	Memory Leak

Advances in AI-Powered Code Security: Next-Level Bug Detection. GitHub talk, https://www.youtube.com/watch?v=nOS56VC0FTQ

RepoAudit: Auditing Code As Human

An autonomous LLM-agent designed for large-scale, repository-level code auditing.

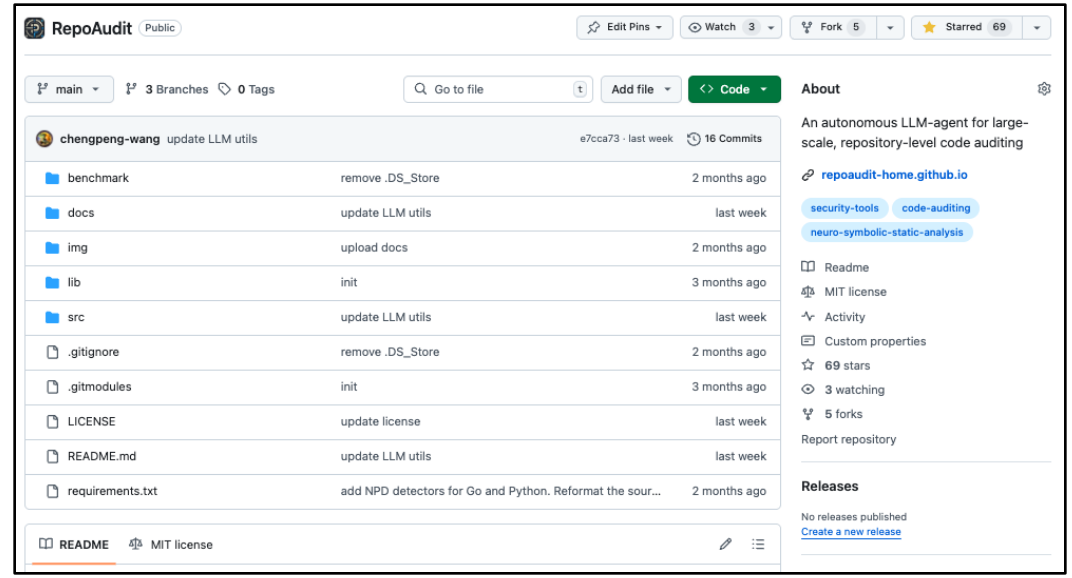
Paper Code

Are you still troubled by code security issues?

Have you ever complained that program testing misses countless bugs?

Have you found static code analysis tools too cumbersome to use, especially when they only support a limited set of bug types and languages like C/C++?

If you face these challenges, RepoAudit is your ultimate lifesaver!



Docs

Check out the following docs to get started with RepoAudit.

You can also refer to the deepwiki page of RepoAudit, which is generated by Devin.

User Guide

Installation

1. Create and activate a conda environment with Python 3.9.18:

```
conda create -n repoaudit python=3.9.18
conda activate repoaudit
```

2. Install the required dependencies:

```
cd RepoAudit
pip install -r requirements.txt
```

Bug Reports

The gallery of bugs discovered by RepoAudit

Fixed: 144 | Confirmed: 111 | Reproduced: 35 | Pending: 82

ID	Project	Lang	Bug Type	Link	Status	Num	Agent	Date
1	zstd	C	Memory Leak	View	Fixed	12	bugscan	2025-6-8
2	Linux	C	Buffer Overflow	View	Pending	3	bugscan	2025-6-7
3	Linux	C	Divide by Zero	View	Fixed	2	bugscan	2025-6-7
4	Linux	C	NPD	View	Pending	3	bugscan	2025-6-6
5	BABEL	C	Functional Bug	View	Reproduced	1	rfcscan	2025-5-30

Advancing Code Auditing with LLMs

GitHub Link

Website Link

Star us on GitHub and explore more on our website