# LLM-Driven Fuzzing

## Automatic Harness Generation for Crypto Libraries

Konstantinos Chousos

July, 2025

Lorem ipsum odor amet, consectetuer adipiscing elit. Habitasse congue tempus erat rhoncus sapien interdum dolor nec. Posuere habitant metus tellus erat eu. Risus ultricies eu rhoncus, conubia euismod convallis commodo per. Nam tellus quisque maximus duis eleifend; arcu aptent. Nisi rutrum primis luctus tortor tempor maecenas. Donec curae cras dolor; malesuada ultricies scelerisque. Molestie class tincidunt quis gravida ut proin. Consequat lacinia arcu justo leo maecenas nunc neque ex. Platea eros ullamcorper nullam rutrum facilisis.

# Preface

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis sagittis posuere ligula sit amet lacinia. Duis dignissim pellentesque magna, rhoncus congue sapien finibus mollis. Ut eu sem laoreet, vehicula ipsum in, convallis erat. Vestibulum magna sem, blandit pulvinar augue sit amet, auctor malesuada sapien. Nullam faucibus leo eget eros hendrerit, non laoreet ipsum lacinia. Curabitur cursus diam elit, non tempus ante volutpat a. Quisque hendrerit blandit purus non fringilla. Integer sit amet elit viverra ante dapibus semper. Vestibulum viverra rutrum enim, at luctus enim posuere eu. Orci varius natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus.

Nunc ac dignissim magna. Vestibulum vitae egestas elit. Proin feugiat leo quis ante condimentum, eu ornare mauris feugiat. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris cursus laoreet ex, dignissim bibendum est posuere iaculis. Suspendisse et maximus elit. In fringilla gravida ornare. Aenean id lectus pulvinar, sagittis felis nec, rutrum risus. Nam vel neque eu arcu blandit fringilla et in quam. Aliquam luctus est sit amet vestibulum eleifend. Phasellus elementum sagittis molestie. Proin tempor lorem arcu, at condimentum purus volutpat eu. Fusce et pellentesque ligula. Pellentesque id tellus at erat luctus fringilla. Suspendisse potenti.

Etiam maximus accumsan gravida. Maecenas at nunc dignissim, euismod enim ac, bibendum ipsum. Maecenas vehicula velit in nisl aliquet ultricies. Nam eget massa interdum, maximus arcu vel, pretium erat. Maecenas sit amet tempor purus, vitae aliquet nunc. Vivamus cursus urna velit, eleifend dictum magna laoreet ut. Duis eu erat mollis, blandit magna id, tincidunt ipsum. Integer massa nibh, commodo eu ex vel, venenatis efficitur ligula. Integer convallis lacus elit, maximus eleifend lacus ornare ac. Vestibulum scelerisque viverra urna id lacinia. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia curae; Aenean eget enim at diam bibendum tincidunt eu non purus. Nullam id magna ultrices, sodales metus viverra, tempus turpis.

# Acknowledgments

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis sagittis posuere ligula sit amet lacinia. Duis dignissim pellentesque magna, rhoncus congue sapien finibus mollis. Ut eu sem laoreet, vehicula ipsum in, convallis erat. Vestibulum magna sem, blandit pulvinar augue sit amet, auctor malesuada sapien. Nullam faucibus leo eget eros hendrerit, non laoreet ipsum lacinia. Curabitur cursus diam elit, non tempus ante volutpat a. Quisque hendrerit blandit purus non fringilla. Integer sit amet elit viverra ante dapibus semper. Vestibulum viverra rutrum enim, at luctus enim posuere eu. Orci varius natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus.

# Table of contents

1 Introduction 1

1.1 Fuzzing . . . . . 1

1.1.1 Fuzzing examples . . . . . 1

1.1.2 Fuzzer engines . . . . . 1

1.2 Large Language Models (LLMs) . . . . . 1

1.2.1 Prompting . . . . . 2

1.3 Neurosymbolic AI . . . . . 2

2 Background Work 3

2.1 Automatic Harnesses . . . . . 3

2.1.1 Google . . . . . 3

2.2 LLM Programming Libraries . . . . . 3

3 LLM-Harness 4

3.1 Scope of Usage . . . . . 4

3.2 Main Library Architecture/Structure . . . . . 4

3.3 GitHub Workflow/Usage . . . . . 4

3.4 Future work/extensions . . . . . 4

4 Conclusion 5

Bibliography 6

Appendices 11

A Failed Techniques 11

# 1 Introduction

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis sagittis posuere ligula sit amet lacinia. Duis dignissim pellentesque magna, rhoncus congue sapien finibus mollis. Ut eu sem laoreet, vehicula ipsum in, convallis erat. Vestibulum magna sem, blandit pulvinar augue sit amet, auctor malesuada sapien. Nullam faucibus leo eget eros hendrerit, non laoreet ipsum lacinia. Curabitur cursus diam elit, non tempus ante volutpat a. Quisque hendrerit blandit purus non fringilla. Integer sit amet elit viverra ante dapibus semper. Vestibulum viverra rutrum enim, at luctus enim posuere eu. Orci varius natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus.

## 1.1 Fuzzing

What is fuzzing [1].

Why fuzz?

### 1.1.1 Fuzzing examples

Heartbleed [2], shellshock [3].

### 1.1.2 Fuzzer engines

C/C++: AFL [4] & AFL++ [4, pp. ++]. LibFuzzer [5].

Python: Atheris [6].

Java, Rust etc...

An example of a fuzz target/harness can be seen in Listing 1.1 [5].

## 1.2 Large Language Models (LLMs)

Transformers [7], 2017–2025. ChatGPT/OpenAI history & context. Claude, Llama (1–3) etc.

**Listing 1.1** A simple function that does something interesting if it receives the input “HI!”.

```
cat << EOF > test_fuzzer.cc
#include <stdint.h>
#include <stddef.h>
extern "C" int LLVMFuzzerTestOneInput(const uint8_t *data, size_t size) {
  if (size > 0 && data[0] == 'H')
    if (size > 1 && data[1] == 'I')
       if (size > 2 && data[2] == '!')
       __builtin_trap();
  return 0;
}
EOF
# Build test_fuzzer.cc with asan and link against libFuzzer.
clang++ -fsanitize=address,fuzzer test_fuzzer.cc
# Run the fuzzer with no corpus.
./a.out
```

### 1.2.1 Prompting

Prompting techniques.

1. Zero-shot.
2. Chain of Thought [8].
3. ReACt [9].
4. Tree of Thoughts [10].

Comparison, strengths weaknesses etc. [11].

## 1.3 Neurosymbolic AI

«««< Updated upstream **TODO** [12]–[17]. ======= ==TODO== [12]–[17]. »»»> Stashed changes

# 2 Background Work

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis sagittis posuere ligula sit amet lacinia. Duis dignissim pellentesque magna, rhoncus congue sapien finibus mollis. Ut eu sem laoreet, vehicula ipsum in, convallis erat. Vestibulum magna sem, blandit pulvinar augue sit amet, auctor malesuada sapien. Nullam faucibus leo eget eros hendrerit, non laoreet ipsum lacinia. Curabitur cursus diam elit, non tempus ante volutpat a. Quisque hendrerit blandit purus non fringilla. Integer sit amet elit viverra ante dapibus semper. Vestibulum viverra rutrum enim, at luctus enim posuere eu. Orci varius natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus.

## 2.1 Automatic Harnesses

Where we are right now. SOTA projects. Similar projects using LLMs in the fuzzing space [18]–[20].

**TODO**.

### 2.1.1 Google

FuzzGen, FUDGE, OSS-Fuzz-Gen [21]–[24].

#### 2.1.1.1 OSS-Fuzz-Gen

Features/caveats. `from_scratch` branch[1].

## 2.2 LLM Programming Libraries

Langchain & LangGraph, LlamaIndex [25]–[27]. DSPy [28].

Comparison, relevance to our usecase.

[1] commit `171aac2`

# 3 LLM-Harness

1. How is it different?
2. What does it offer?

## 3.1 Scope of Usage

1. In what contexts does it work?
2. Prerequisites

## 3.2 Main Library Architecture/Structure

## 3.3 GitHub Workflow/Usage

## 3.4 Future work/extensions

1. Build system
2. General *localization* problem

# 4 Conclusion

1. What works/what doesn’t
2. More general future steps for the field in general
3. Recap

# Bibliography

[1] V. J. M. Manes, H. Han, C. Han, *et al.* "The Art, Science, and Engineering of Fuzzing: A Survey." arXiv: 1812.00140 [cs]. (Apr. 7, 2019), [Online]. Available: http://arxiv.org/abs/1812.00140, pre-published.

[2] "Heartbleed Bug." (), [Online]. Available: https://heartbleed.com/.

[3] C. Meyer and J. Schwenk. "Lessons Learned From Previous SSL/TLS Attacks - A Brief Chronology Of Attacks And Weaknesses." (2013), [Online]. Available: https://eprint.iacr.org/2013/049, pre-published.

[4] "American fuzzy lop." (), [Online]. Available: https://lcamtuf.coredump.cx/afl/.

[5] "libFuzzer – a library for coverage-guided fuzz testing. — LLVM 21.0.0git documentation." (), [Online]. Available: https://llvm.org/docs/LibFuzzer.html.

[6] *Google/atheris*, Google, Apr. 9, 2025. [Online]. Available: https://github.com/google/atheris.

[7] A. Vaswani, N. Shazeer, N. Parmar, *et al.* "Attention Is All You Need." arXiv: 1706.03762 [cs]. (Aug. 1, 2023), [Online]. Available: http://arxiv.org/abs/1706.03762, pre-published.

[8] J. Wei, X. Wang, D. Schuurmans, *et al.* "Chain-of-Thought Prompting Elicits Reasoning in Large Language Models." arXiv: 2201.11903 [cs]. (Jan. 10, 2023), [Online]. Available: http://arxiv.org/abs/2201.11903, pre-published.

[9] S. Yao, J. Zhao, D. Yu, *et al.* "ReAct: Synergizing Reasoning and Acting in Language Models." arXiv: 2210.03629. (Mar. 10, 2023), [Online]. Available: http://arxiv.org/abs/2210.03629, pre-published.

[10] S. Yao, D. Yu, J. Zhao, *et al.* "Tree of Thoughts: Deliberate Problem Solving with Large Language Models." arXiv: 2305.10601 [cs]. (Dec. 3, 2023), [Online]. Available: http://arxiv.org/abs/2305.10601, pre-published.

[11] P. Laban, H. Hayashi, Y. Zhou, and J. Neville. "LLMs Get Lost In Multi-Turn Conversation." arXiv: 2505.06120 [cs]. (May 9, 2025), [Online]. Available: http://arxiv.org/abs/2505.06120, pre-published.

[12] D. Ganguly, S. Iyengar, V. Chaudhary, and S. Kalyanaraman. "Proof of Thought : Neurosymbolic Program Synthesis allows Robust and Interpretable Reasoning." arXiv: 2409.17270. (Sep. 25, 2024), [Online]. Available: http://arxiv.org/abs/2409.17270, pre-published.

[13] A. d'Avila Garcez and L. C. Lamb. "Neurosymbolic AI: The 3rd Wave." arXiv: 2012.05876. (Dec. 16, 2020), [Online]. Available: http://arxiv.org/abs/2012.05876, pre-published.

[14] M. Gaur and A. Sheth. "Building Trustworthy NeuroSymbolic AI Systems: Consistency, Reliability, Explainability, and Safety." arXiv: 2312.06798. (Dec. 5, 2023), [Online]. Available: http://arxiv.org/abs/2312.06798, pre-published.

[15] G. Grov, J. Halvorsen, M. W. Eckhoff, B. J. Hansen, M. Eian, and V. Mavroeidis. “On the use of neurosymbolic AI for defending against cyber attacks.” arXiv: 2408.04996. (Aug. 9, 2024), [Online]. Available: http://arxiv.org/abs/2408.04996, pre-published.

[16] A. Sheth, K. Roy, and M. Gaur. “Neurosymbolic AI – Why, What, and How.” arXiv: 2305.00813 [cs]. (May 1, 2023), [Online]. Available: http://arxiv.org/abs/2305.00813, pre-published.

[17] D. Tilwani, R. Venkataramanan, and A. P. Sheth. “Neurosymbolic AI approach to Attribution in Large Language Models.” arXiv: 2410.03726. (Sep. 30, 2024), [Online]. Available: http://arxiv.org/abs/2410.03726, pre-published.

[18] Y. Deng, C. S. Xia, C. Yang, S. D. Zhang, S. Yang, and L. Zhang. “Large Language Models are Edge-Case Fuzzers: Testing Deep Learning Libraries via FuzzGPT.” arXiv: 2304.02014 [cs]. (Apr. 4, 2023), [Online]. Available: http://arxiv.org/abs/2304.02014, pre-published.

[19] Y. Deng, C. S. Xia, H. Peng, C. Yang, and L. Zhang, “Large Language Models Are Zero-Shot Fuzzers: Fuzzing Deep-Learning Libraries via Large Language Models,” in *Proceedings of the 32nd ACM SIGSOFT International Symposium on Software Testing and Analysis*, ser. ISSTA 2023, New York, NY, USA: Association for Computing Machinery, Jul. 13, 2023, pp. 423–435, ISBN: 979-8-4007-0221-1. DOI: 10.1145/3597926.3598067. [Online]. Available: https://dl.acm.org/doi/10.1145/3597926.3598067.

[20] Z. Li, S. Dutta, and M. Naik. “IRIS: LLM-Assisted Static Analysis for Detecting Security Vulnerabilities.” arXiv: 2405.17238 [cs]. (Apr. 6, 2025), [Online]. Available: http://arxiv.org/abs/2405.17238, pre-published.

[21] K. Ispoglou, D. Austin, V. Mohan, and M. Payer, “FuzzGen: Automatic fuzzer generation,” in *29th USENIX Security Symposium (USENIX Security 20)*, 2020, pp. 2271–2287. [Online]. Available: https://www.usenix.org/conference/usenixsecurity20/presentation/ispoglou.

[22] D. Babić, S. Bucur, Y. Chen, *et al.*, “FUDGE: Fuzz driver generation at scale,” in *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, Tallinn Estonia: ACM, Aug. 12, 2019, pp. 975–985, ISBN: 978-1-4503-5572-8. DOI: 10.1145/3338906.3340456. [Online]. Available: https://dl.acm.org/doi/10.1145/3338906.3340456.

[23] A. Arya, O. Chang, J. Metzman, K. Serebryany, and D. Liu, *OSS-Fuzz*, Apr. 8, 2025. [Online]. Available: https://github.com/google/oss-fuzz.

[24] D. Liu, O. Chang, J. metzman, M. Sablotny, and M. Maruseac, *OSS-fuzz-gen: Automated fuzz target generation*, version https://github.com/google/oss-fuzz-gen/tree/v1.0, May 2024. [Online]. Available: https://github.com/google/oss-fuzz-gen.

[25] H. Chase, *LangChain*, Oct. 2022. [Online]. Available: https://github.com/langchain-ai/langchain.

[26] *Langchain-ai/langgraph*, LangChain, May 21, 2025. [Online]. Available: https://github.com/langchain-ai/langgraph.

[27] J. Liu, *LlamaIndex*, Nov. 2022. DOI: 10.5281/zenodo.1234. [Online]. Available: https://github.com/jerryjliu/llama_index.

[28] O. Khattab, A. Singhvi, P. Maheshwari, *et al.* “DSPy: Compiling Declarative Language Model Calls into Self-Improving Pipelines.” arXiv: 2310.03714 [cs]. (Oct. 5, 2023), [Online]. Available: http://arxiv.org/abs/2310.03714, pre-published.

[29] *Google/clusterfuzz*, Google, Apr. 9, 2025. [Online]. Available: https://github.com/google/clusterfuzz.

[30] M. Heuse, H. Eißfeldt, A. Fioraldi, and D. Maier, *AFL++*, version 4.00c, Jan. 2022. [Online]. Available: https://github.com/AFLplusplus/AFLplusplus.

[31] A. Fioraldi, D. Maier, H. Eißfeldt, and M. Heuse, "AFL++: Combining incremental steps of fuzzing research," in *14th USENIX Workshop on Offensive Technologies (WOOT 20)*, USENIX Association, Aug. 2020.

[32] R. Anderson, "Why cryptosystems fail," in *Proceedings of the 1st ACM Conference on Computer and Communications Security - CCS '93*, Fairfax, Virginia, United States: ACM Press, 1993, pp. 215–227, ISBN: 978-0-89791-629-5. DOI: 10.1145/168588.168615. [Online]. Available: http://portal.acm.org/citation.cfm?doid=168588.168615.

[33] Y. Cheng, H. J. Kang, L. K. Shar, *et al.* "Towards Reliable LLM-Driven Fuzz Testing: Vision and Road Ahead." arXiv: 2503.00795 [cs]. (Mar. 2, 2025), [Online]. Available: http://arxiv.org/abs/2503.00795, pre-published.

[34] W. Gao, V.-T. Pham, D. Liu, O. Chang, T. Murray, and B. I. Rubinstein, "Beyond the Coverage Plateau: A Comprehensive Study of Fuzz Blockers (Registered Report)," in *Proceedings of the 2nd International Fuzzing Workshop*, ser. FUZZING 2023, New York, NY, USA: Association for Computing Machinery, Jul. 17, 2023, pp. 47–55, ISBN: 979-8-4007-0247-1. DOI: 10.1145/3605157.3605177. [Online]. Available: https://dl.acm.org/doi/10.1145/3605157.3605177.

[35] Y. Gao, Y. Xiong, X. Gao, *et al.* "Retrieval-Augmented Generation for Large Language Models: A Survey." arXiv: 2312.10997 [cs]. (Mar. 27, 2024), [Online]. Available: http://arxiv.org/abs/2312.10997, pre-published.

[36] L. Gazzola, D. Micucci, and L. Mariani, "Automatic Software Repair: A Survey," *IEEE Transactions on Software Engineering*, vol. 45, no. 1, pp. 34–67, Jan. 2019, ISSN: 1939-3520. DOI: 10.1109/TSE.2017.2755013. [Online]. Available: https://ieeexplore.ieee.org/document/8089448/.

[37] D. Giannone. "Demystifying AI Agents: ReAct-Style Agents vs Agentic Workflows," Medium. (Feb. 9, 2025), [Online]. Available: https://medium.com/@DanGiannone/demystifying-ai-agents-react-style-agents-vs-agentic-workflows-cedca7e26471.

[38] A. Herrera, H. Gunadi, S. Magrath, M. Norrish, M. Payer, and A. L. Hosking, "Seed selection for successful fuzzing," in *Proceedings of the 30th ACM SIGSOFT International Symposium on Software Testing and Analysis*, Virtual Denmark: ACM, Jul. 11, 2021, pp. 230–243, ISBN: 978-1-4503-8459-9. DOI: 10.1145/3460319.3464795. [Online]. Available: https://dl.acm.org/doi/10.1145/3460319.3464795.

[39] L. Huang, P. Zhao, H. Chen, and L. Ma. "Large language models based fuzzing techniques: A survey." (2024), [Online]. Available: https://arxiv.org/abs/2402.00350, pre-published.

[40] R. I. T. Jensen, V. Tawosi, and S. Alamir. "Software Vulnerability and Functionality Assessment using LLMs." arXiv: 2403.08429. (Mar. 13, 2024), [Online]. Available: http://arxiv.org/abs/2403.08429, pre-published.

[41] S. Y. Kim, Z. Fan, Y. Noller, and A. Roychoudhury. "Codexity: Secure AI-assisted Code Generation." version 1. arXiv: 2405.03927. (May 7, 2024), [Online]. Available: http://arxiv.org/abs/2405.03927, pre-published.

[42] C. Cadar, D. Dunbar, and D. Engler, “KLEE: Unassisted and Automatic Generation of High-Coverage Tests for Complex Systems Programs,” presented at the USENIX Symposium on Operating Systems Design and Implementation, Dec. 8, 2008. [Online]. Available: https://www.semanticscholar.org/paper/KLEE%3A-Unassisted-and-Automatic-Generation-of-Tests-Cadar-Dunbar/0b93657965e506dfbd56fbc1c1d4b9666b1d01c8.

[43] D. Lazar, H. Chen, X. Wang, and N. Zeldovich, “Why does cryptographic software fail? a case study and open problems,” in *Proceedings of 5th Asia-Pacific Workshop on Systems*, ser. APSys ’14, New York, NY, USA: Association for Computing Machinery, Jun. 25, 2014, pp. 1–7, ISBN: 978-1-4503-3024-4. DOI: 10.1145/2637166.2637237. [Online]. Available: https://doi.org/10.1145/2637166.2637237.

[44] H.-P. H. Lee, A. Sarkar, L. Tankelevitch, *et al.*, “The Impact of Generative AI on Critical Thinking: Self-Reported Reductions in Cognitive Effort and Confidence Effects From a Survey of Knowledge Workers,” 2025. [Online]. Available: https://hankhplee.com/papers/genai_critical_thinking.pdf.

[45] C. Le Goues, S. Forrest, and W. Weimer, “Current challenges in automatic software repair,” *Software Quality Journal*, vol. 21, no. 3, pp. 421–443, Sep. 2013, ISSN: 0963-9314, 1573-1367. DOI: 10.1007/s11219-013-9208-0. [Online]. Available: http://link.springer.com/10.1007/s11219-013-9208-0.

[46] H. Li, “Language models: Past, present, and future,” *Commun. ACM*, vol. 65, no. 7, pp. 56–63, Jun. 21, 2022, ISSN: 0001-0782. DOI: 10.1145/3490443. [Online]. Available: https://dl.acm.org/doi/10.1145/3490443.

[47] J. C. R. Licklider, “Man-Computer Symbiosis,” *IRE Transactions on Human Factors in Electronics*, vol. HFE-1, no. 1, pp. 4–11, Mar. 1960, ISSN: 2168-2836. DOI: 10.1109/THFE2.1960.4503259. [Online]. Available: https://ieeexplore.ieee.org/document/4503259.

[48] J. Liu, S. Lee, E. Losiouk, and M. Böhme. “Can LLM Generate Regression Tests for Software Commits?” arXiv: 2501.11086 [cs]. (Jan. 19, 2025), [Online]. Available: http://arxiv.org/abs/2501.11086, pre-published.

[49] Y. Lyu, Y. Xie, P. Chen, and H. Chen. “Prompt Fuzzing for Fuzz Driver Generation.” arXiv: 2312.17677 [cs]. (May 29, 2024), [Online]. Available: http://arxiv.org/abs/2312.17677, pre-published.

[50] N. Nethercote and J. Seward, “Valgrind: A framework for heavyweight dynamic binary instrumentation,” *SIGPLAN Not.*, vol. 42, no. 6, pp. 89–100, Jun. 10, 2007, ISSN: 0362-1340. DOI: 10.1145/1273442.1250746. [Online]. Available: https://doi.org/10.1145/1273442.1250746.

[51] OSS-Fuzz Maintainers. “Introducing LLM-based harness synthesis for unfuzzed projects,” OSS-Fuzz blog. (May 27, 2024), [Online]. Available: https://blog.oss-fuzz.com/posts/introducing-llm-based-harness-synthesis-for-unfuzzed-projects/.

[52] N. Perry, M. Srivastava, D. Kumar, and D. Boneh. “Do Users Write More Insecure Code with AI Assistants?” arXiv: 2211.03622. (Dec. 18, 2023), [Online]. Available: http://arxiv.org/abs/2211.03622, pre-published.

[53] D. Wang, G. Zhou, L. Chen, D. Li, and Y. Miao. “ProphetFuzz: Fully Automated Prediction and Fuzzing of High-Risk Option Combinations with Only Documentation via Large Language Model.” arXiv: 2409.00922 [cs]. (Sep. 1, 2024), [Online]. Available: http://arxiv.org/abs/2409.00922, pre-published.

[54] T. Simonite, “This Bot Hunts Software Bugs for the Pentagon,” *Wired*, Jun. 1, 2020, ISSN: 1059-1028. [Online]. Available: https://www.wired.com/story/bot-hunts-software-bugs-pentagon/.

[55] I. Tzachristas. “Creating an LLM-based AI-agent: A high-level methodology towards enhancing LLMs with APIs.” version 2. arXiv: 2412.13233 [cs]. (Dec. 21, 2024), [Online]. Available: http://arxiv.org/abs/2412.13233, pre-published.

[56] A. Zebaze, B. Sagot, and R. Bawden. “Tree of Problems: Improving structured problem solving with compositionality.” arXiv: 2410.06634. (Oct. 9, 2024), [Online]. Available: http://arxiv.org/abs/2410.06634, pre-published.

[57] C. Zhang, Y. Zheng, M. Bai, *et al.*, “How Effective Are They? Exploring Large Language Model Based Fuzz Driver Generation,” in *Proceedings of the 33rd ACM SIGSOFT International Symposium on Software Testing and Analysis*, Sep. 11, 2024, pp. 1223–1235. DOI: 10.1145/3650212.3680355. arXiv: 2307.12469 [cs]. [Online]. Available: http://arxiv.org/abs/2307.12469.

[58] C. Zhang, Y. Zheng, M. Bai, *et al.*, “How Effective Are They? Exploring Large Language Model Based Fuzz Driver Generation,” in *Proceedings of the 33rd ACM SIGSOFT International Symposium on Software Testing and Analysis*, ser. ISSTA 2024, New York, NY, USA: Association for Computing Machinery, Sep. 11, 2024, pp. 1223–1235, ISBN: 979-8-4007-0612-7. DOI: 10.1145/3650212.3680355. [Online]. Available: https://dl.acm.org/doi/10.1145/3650212.3680355.

[59] K. Zhang, S. Wang, J. Han, *et al.* “Your Fix Is My Exploit: Enabling Comprehensive DL Library API Fuzzing with Large Language Models.” arXiv: 2501.04312 [cs]. (Jan. 8, 2025), [Online]. Available: http://arxiv.org/abs/2501.04312, pre-published.

[60] P. Y. Zhong, H. He, O. Khattab, C. Potts, M. Zaharia, and H. Miller. “A Guide to Large Language Model Abstractions,” Two Sigma. (Jan. 16, 2024), [Online]. Available: https://www.twosigma.com/articles/a-guide-to-large-language-model-abstractions/.

[61] A. Zibaeirad and M. Vieira. “Reasoning with LLMs for Zero-Shot Vulnerability Detection.” arXiv: 2503.17885 [cs]. (Mar. 22, 2025), [Online]. Available: http://arxiv.org/abs/2503.17885, pre-published.

[62] “How to Prevent the next Heartbleed.” (), [Online]. Available: https://dwheeler.com/essays/heartbleed.html.

[63] “AI-Powered Fuzzing: Breaking the Bug Hunting Barrier,” Google Online Security Blog. (), [Online]. Available: https://security.googleblog.com/2023/08/ai-powered-fuzzing-breaking-bug-hunting.html.

[64] “OSS-Fuzz Documentation,” OSS-Fuzz. (), [Online]. Available: https://google.github.io/oss-fuzz/.

# A Failed Techniques

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis sagittis posuere ligula sit amet lacinia. Duis dignissim pellentesque magna, rhoncus congue sapien finibus mollis. Ut eu sem laoreet, vehicula ipsum in, convallis erat. Vestibulum magna sem, blandit pulvinar augue sit amet, auctor malesuada sapien. Nullam faucibus leo eget eros hendrerit, non laoreet ipsum lacinia. Curabitur cursus diam elit, non tempus ante volutpat a. Quisque hendrerit blandit purus non fringilla. Integer sit amet elit viverra ante dapibus semper. Vestibulum viverra rutrum enim, at luctus enim posuere eu. Orci varius natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus.

Nunc ac dignissim magna. Vestibulum vitae egestas elit. Proin feugiat leo quis ante condimentum, eu ornare mauris feugiat. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris cursus laoreet ex, dignissim bibendum est posuere iaculis. Suspendisse et maximus elit. In fringilla gravida ornare. Aenean id lectus pulvinar, sagittis felis nec, rutrum risus. Nam vel neque eu arcu blandit fringilla et in quam. Aliquam luctus est sit amet vestibulum eleifend. Phasellus elementum sagittis molestie. Proin tempor lorem arcu, at condimentum purus volutpat eu. Fusce et pellentesque ligula. Pellentesque id tellus at erat luctus fringilla. Suspendisse potenti.

Etiam maximus accumsan gravida. Maecenas at nunc dignissim, euismod enim ac, bibendum ipsum. Maecenas vehicula velit in nisl aliquet ultricies. Nam eget massa interdum, maximus arcu vel, pretium erat. Maecenas sit amet tempor purus, vitae aliquet nunc. Vivamus cursus urna velit, eleifend dictum magna laoreet ut. Duis eu erat mollis, blandit magna id, tincidunt ipsum. Integer massa nibh, commodo eu ex vel, venenatis efficitur ligula. Integer convallis lacus elit, maximus eleifend lacus ornare ac. Vestibulum scelerisque viverra urna id lacinia. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia curae; Aenean eget enim at diam bibendum tincidunt eu non purus. Nullam id magna ultrices, sodales metus viverra, tempus turpis.

Duis ornare ex ac iaculis pretium. Maecenas sagittis odio id erat pharetra, sit amet consectetur quam sollicitudin. Vivamus pharetra quam purus, nec sagittis risus pretium at. Nullam feugiat, turpis ac accumsan interdum, sem tellus blandit neque, id vulputate diam quam semper nisl. Donec sit amet enim at neque porttitor aliquet. Phasellus facilisis nulla eget placerat eleifend. Vestibulum non egestas eros, eget lobortis ipsum. Nulla rutrum massa eget enim aliquam, id porttitor erat luctus. Nunc sagittis quis eros eu sagittis. Pellentesque dictum, erat at pellentesque sollicitudin, justo augue pulvinar metus, quis rutrum est mi nec felis. Vestibulum efficitur mi lorem, at elementum purus tincidunt a. Aliquam finibus enim magna, vitae pellentesque erat faucibus at. Nulla mauris tellus, imperdiet id lobortis et, dignissim condimentum ipsum. Morbi nulla orci, varius at aliquet sed, facilisis id tortor. Donec ut urna nisi.

Aenean placerat luctus tortor vitae molestie. Nulla at aliquet nulla. Sed efficitur tellus orci, sed fringilla lectus laoreet eget. Vivamus maximus quam sit amet arcu dignissim, sed accumsan massa ullamcorper. Sed iaculis tincidunt feugiat. Nulla in est at nunc ultricies dictum ut vitae nunc. Aenean convallis vel diam at malesuada. Suspendisse arcu libero, vehicula tempus ultrices a, placerat sit amet tortor. Sed dictum id nulla commodo mattis. Aliquam mollis, nunc eu tristique faucibus, purus lacus tincidunt nulla, ac pretium lorem nunc ut enim. Curabitur eget mattis nisl, vitae sodales augue. Nam felis massa, bibendum sit amet nulla vel, vulputate rutrum lacus. Aenean convallis odio pharetra nulla mattis consequat.

Ut ut condimentum augue, nec eleifend nisl. Sed facilisis egestas odio ac pretium. Pellentesque consequat magna sed venenatis sagittis. Vivamus feugiat lobortis magna vitae accumsan. Pellentesque euismod malesuada hendrerit. Ut non mauris non arcu condimentum sodales vitae vitae dolor. Nullam dapibus, velit eget lacinia rutrum, ipsum justo malesuada odio, et lobortis sapien magna vel lacus. Nulla purus neque, hendrerit non malesuada eget, mattis vel erat. Suspendisse potenti.

Nullam dapibus cursus dolor sit amet consequat. Nulla facilisi. Curabitur vel nulla non magna lacinia tincidunt. Duis porttitor quam leo, et blandit velit efficitur ut. Etiam auctor tincidunt porttitor. Phasellus sed accumsan mi. Fusce ut erat dui. Suspendisse eu augue eget turpis condimentum finibus eu non lorem. Donec finibus eros eu ante condimentum, sed pharetra sapien sagittis. Phasellus non dolor ac ante mollis auctor nec et sapien. Pellentesque vulputate at nisi eu tincidunt. Vestibulum at dolor aliquam, hendrerit purus eu, eleifend massa. Morbi consectetur eros id tincidunt gravida. Fusce ut enim quis orci hendrerit lacinia sed vitae enim.