CARIAD

Master Thesis: AI Usage in CI/CD/CT Pipelines for Compute Platforms in Automotives

We transform automotive mobility

CARIAD
A VOLKSWAGEN GROUP COMPANY

Agenda

// Introduction

// Problem Statement

// Overview

// Key Differences & Challenges

// Method & Process

// Initial Results

// Expected Outcomes

// Next Steps

// Conlcusion

Introduction

Introduction

- A modern car can have around 100 million lines of code, and this is expected to increase to around 300 million by 2030
- Manual fuzz driver creation is time-intensive and requires deep expertise
- Automotive software complexity demands continuous security validation
- CI/CD/CT pipeline integration needs automation for scalability

How can Large Language Models automate and enhance security testing in automotive CI/CD pipelines?

Problem Statement

Problem Statement

- Current white-box fuzzing & testing are manual or slow to scale
- Vulnerabilities may slip through scheduled CI runs due to time constraints
- Need an AI-guided approach integrated into CI/CD/CT to

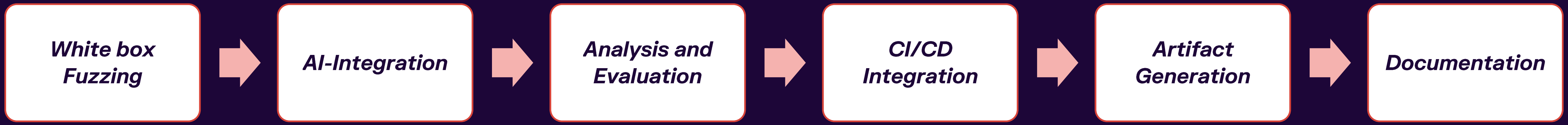
– boost path coverage and

– auto-generate actionable test artifacts

"What if we could make every developer a security testing expert through AI assistance?"

Overview

Overview



White box Fuzzing

AI-Integration

Analysis and Evaluation

CI/CD Integration

Artifact Generation

Documentation

Key Differences

Key Differences

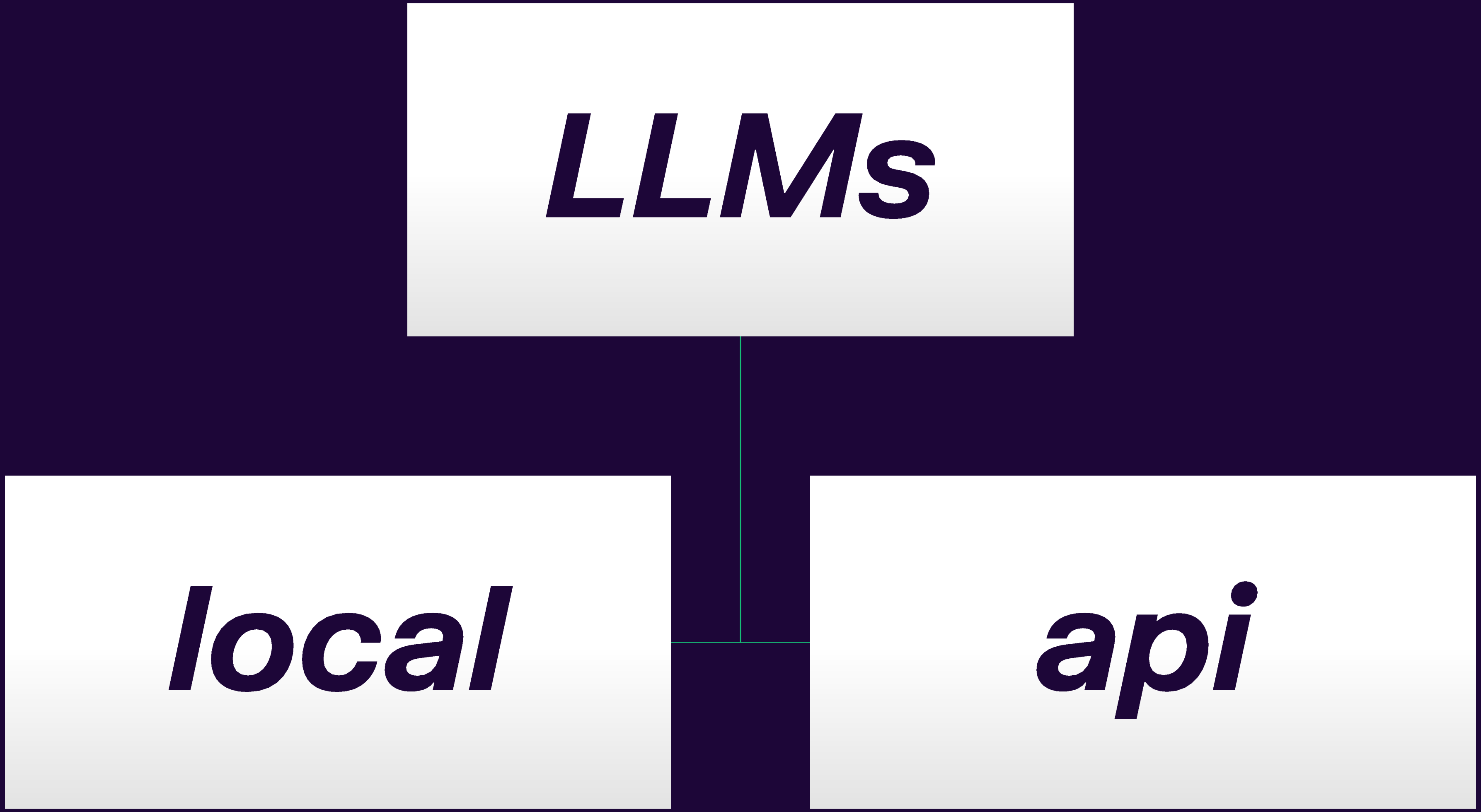
Normal Fuzzing	LLM Based Fuzzing
Random mutation of seed inputs	Content-aware, AI-driven input generation
Brute-force, random	Reinforcement learning, symbolic execution
Static, predefined input mutations	Dynamic, learns from previous tests (Continuous Fuzzing)

Challanges

1. Hallucinations
2. Limited memory for large codebases
3. Bias and Training Data Issues
4. Inconsistent Quality and Reliability
5. Security and Privacy Risks
6. Overfitting and Generalization Issues
7. Prompt Engineering Sensitivity
8. Evaluation and Validation Challenges

Method

Method

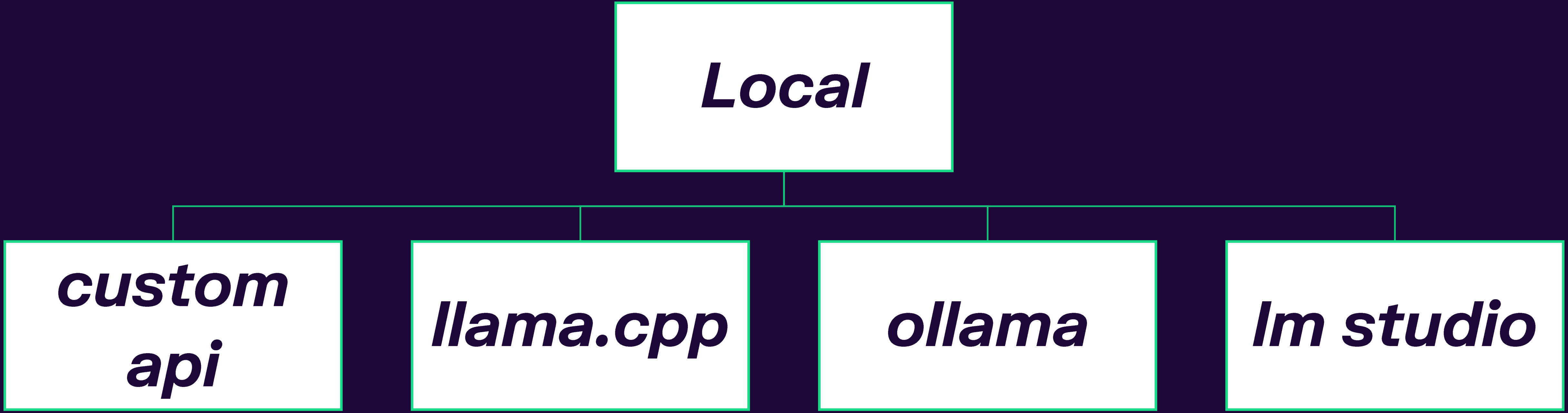


LLMs

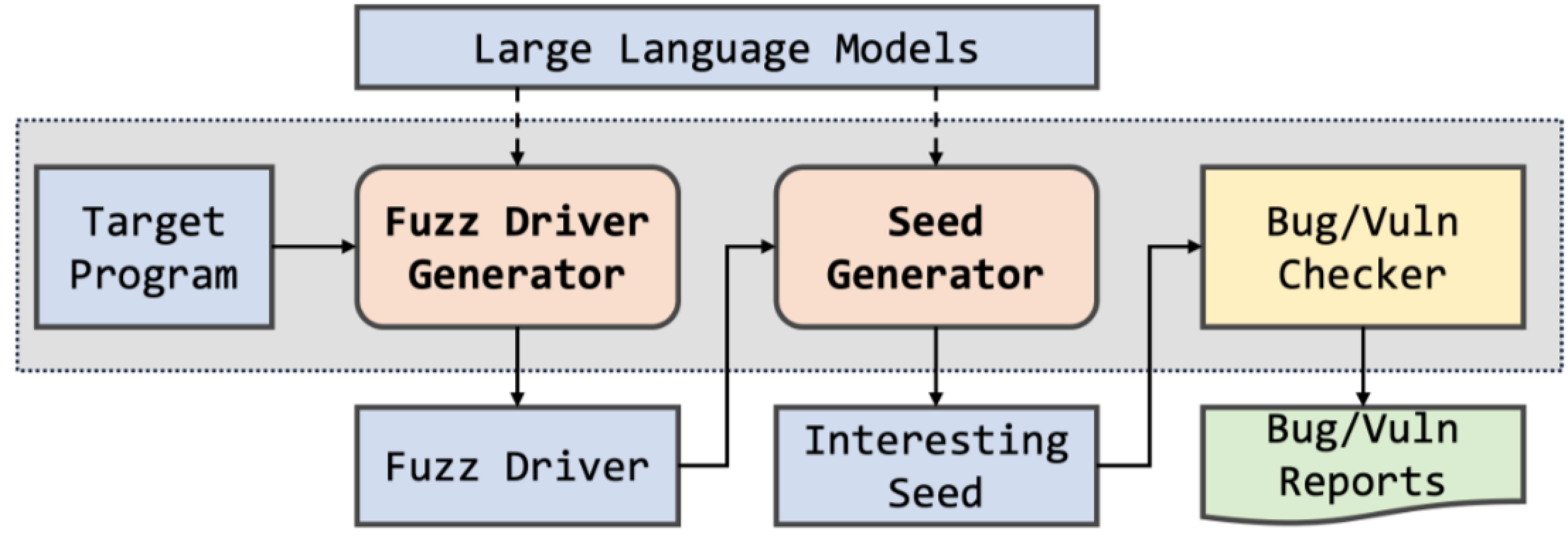
local

api

Method



Process

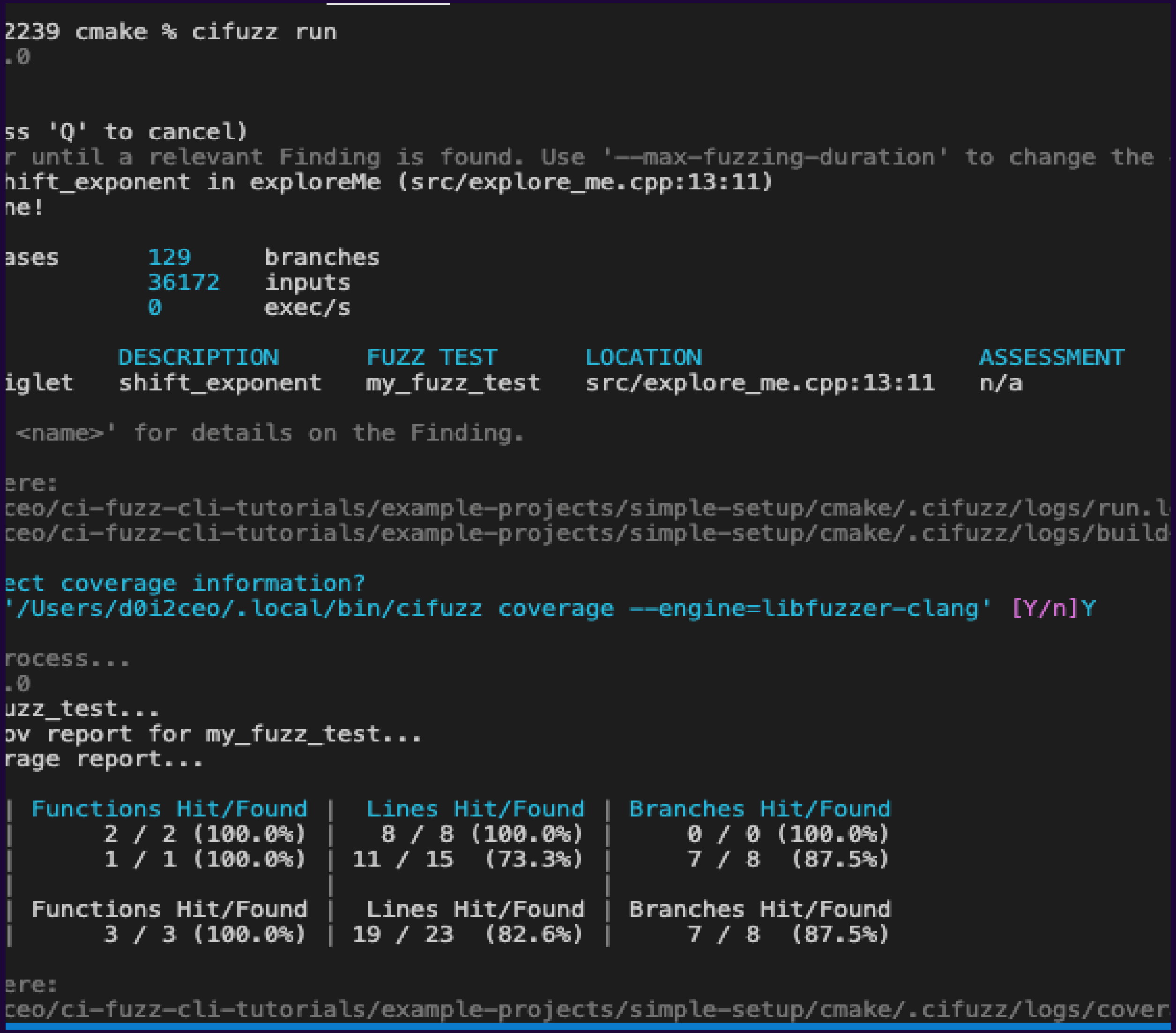

Large Language Models
Target Program
Fuzz Driver Generator
Seed Generator
Bug/Vuln Checker
Fuzz Driver
Interesting Seed
Bug/Vuln Reports

Initial Results

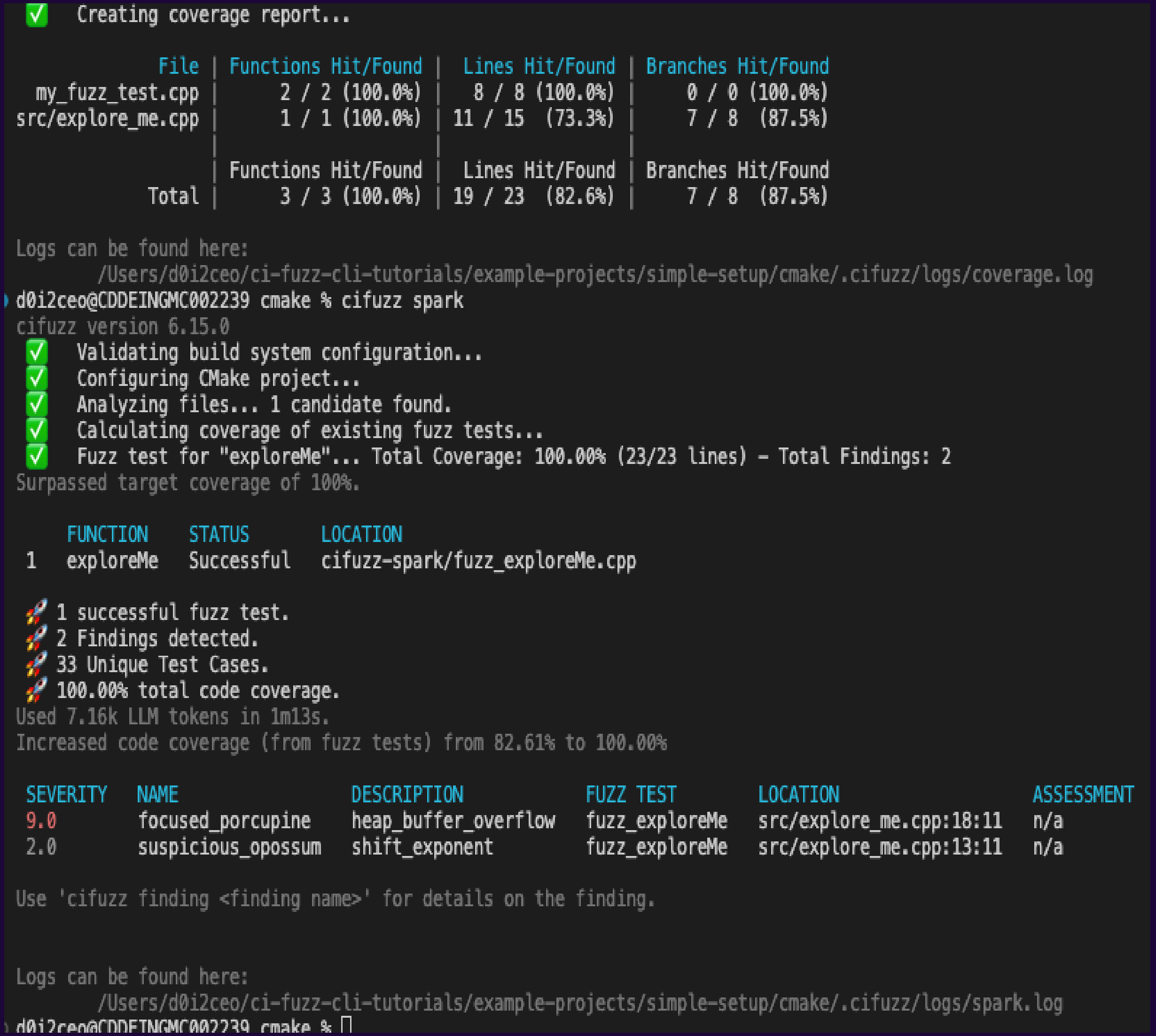
Literature Review

1. When Fuzzing Meets LLMs Challenges and Opportunities
2. An Empirical Study of OSS-Fuzz Bugs
3. Towards LLMs Guided Kernel Direct Fuzzing
4. LLMs for fuzzing parsers

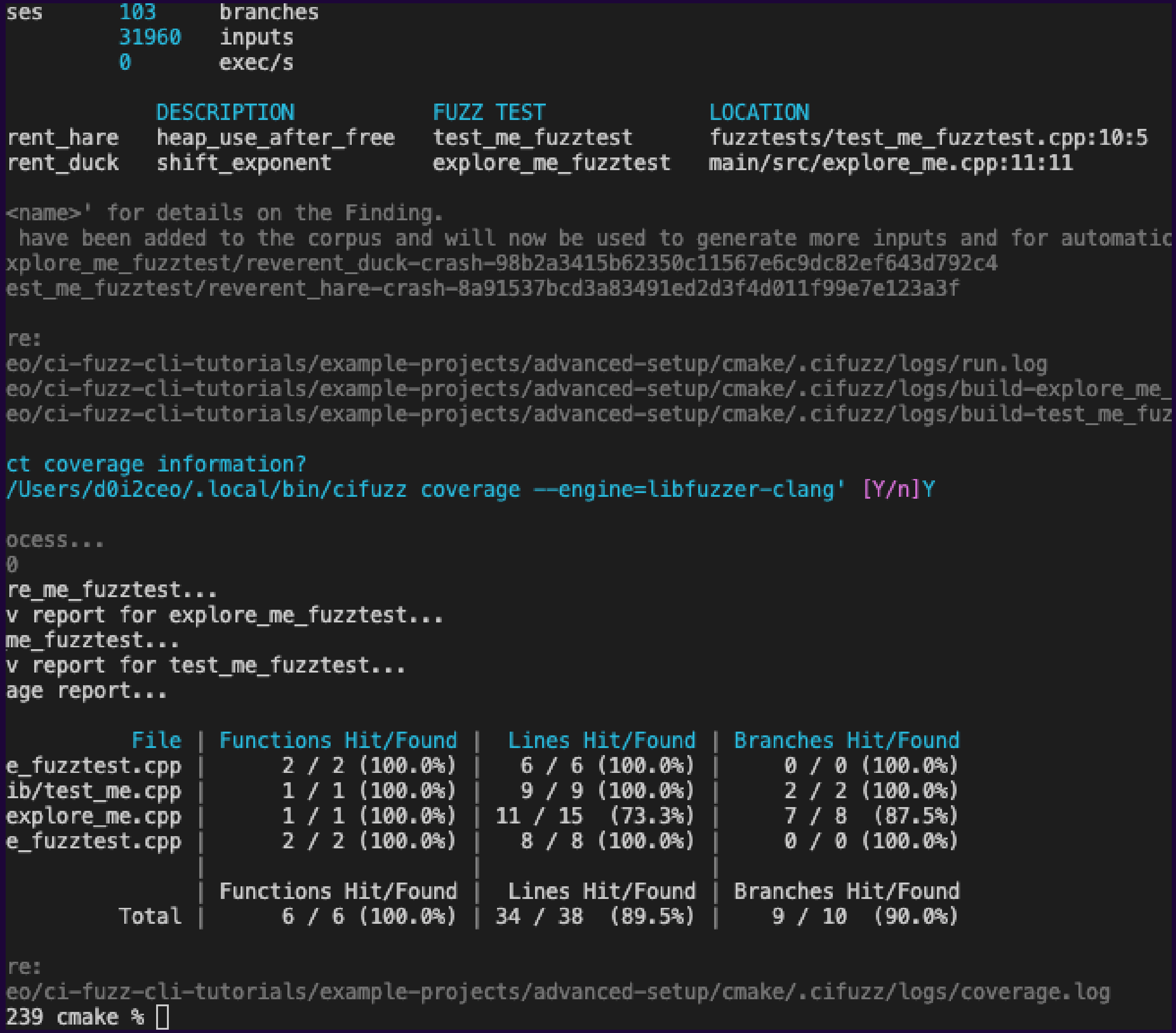
Literature Review



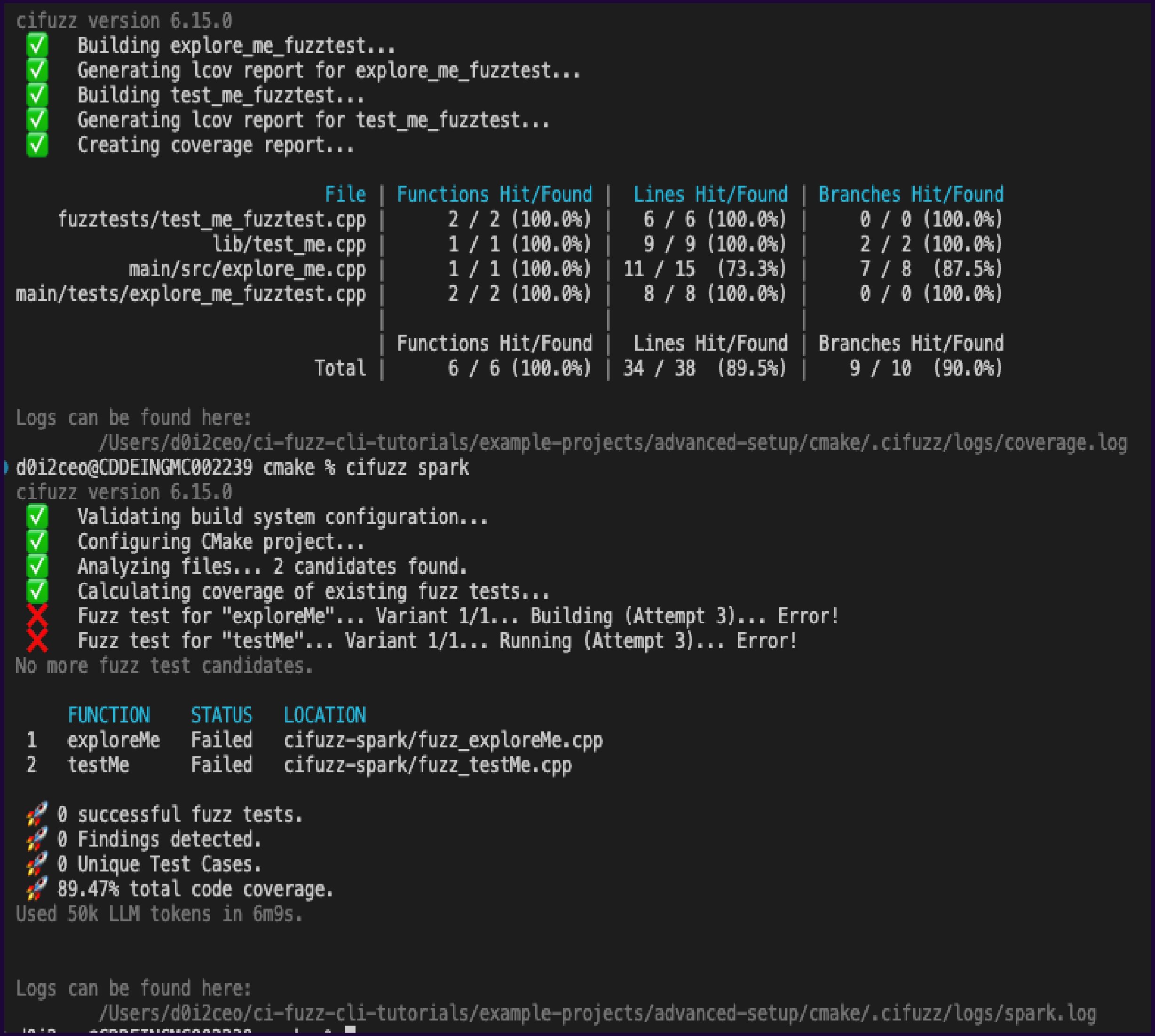
```
2239 cmake % cifuzz run
.0


ss 'Q' to cancel)
r until a relevant Finding is found. Use '--max-fuzzing-duration' to change the
hift_exponent in exploreMe (src/explore_me.cpp:13:11)
ne!

ases       129     branches
           36172   inputs
           0       exec/s

        DESCRIPTION      FUZZ TEST      LOCATION               ASSESSMENT
iglet   shift_exponent   my_fuzz_test   src/explore_me.cpp:13:11   n/a

 <name>' for details on the Finding.

ere:
ceo/ci-fuzz-cli-tutorials/example-projects/simple-setup/cmake/.cifuzz/logs/run.l
ceo/ci-fuzz-cli-tutorials/example-projects/simple-setup/cmake/.cifuzz/logs/build

ect coverage information?
'/Users/d0i2ceo/.local/bin/cifuzz coverage --engine=libfuzzer-clang' [Y/n]Y

rocess...
.0
uzz_test...
ov report for my_fuzz_test...
rage report...

| Functions Hit/Found |  Lines Hit/Found | Branches Hit/Found
|      2 / 2 (100.0%) |   8 / 8 (100.0%) |     0 / 0 (100.0%)
|      1 / 1 (100.0%) | 11 / 15  (73.3%) |     7 / 8  (87.5%)
|                     |                  |
| Functions Hit/Found |  Lines Hit/Found | Branches Hit/Found
|      3 / 3 (100.0%) | 19 / 23  (82.6%) |     7 / 8  (87.5%)

ere:
ceo/ci-fuzz-cli-tutorials/example-projects/simple-setup/cmake/.cifuzz/logs/cover
```



```
✅  Creating coverage report...

             File | Functions Hit/Found |  Lines Hit/Found | Branches Hit/Found
  my_fuzz_test.cpp |      2 / 2 (100.0%) |   8 / 8 (100.0%) |     0 / 0 (100.0%)
src/explore_me.cpp |      1 / 1 (100.0%) | 11 / 15  (73.3%) |     7 / 8  (87.5%)
                   |                     |                  |
                   | Functions Hit/Found |  Lines Hit/Found | Branches Hit/Found
             Total |      3 / 3 (100.0%) | 19 / 23  (82.6%) |     7 / 8  (87.5%)

Logs can be found here:
        /Users/d0i2ceo/ci-fuzz-cli-tutorials/example-projects/simple-setup/cmake/.cifuzz/logs/coverage.log
d0i2ceo@CDDEINGMC002239 cmake % cifuzz spark
cifuzz version 6.15.0
✅  Validating build system configuration...
✅  Configuring CMake project...
✅  Analyzing files... 1 candidate found.
✅  Calculating coverage of existing fuzz tests...
✅  Fuzz test for "exploreMe"... Total Coverage: 100.00% (23/23 lines) - Total Findings: 2
Surpassed target coverage of 100%.

    FUNCTION   STATUS       LOCATION
 1  exploreMe  Successful   cifuzz-spark/fuzz_exploreMe.cpp

🚀 1 successful fuzz test.
🚀 2 Findings detected.
🚀 33 Unique Test Cases.
🚀 100.00% total code coverage.
Used 7.16k LLM tokens in 1m13s.
Increased code coverage (from fuzz tests) from 82.61% to 100.00%

 SEVERITY  NAME                DESCRIPTION           FUZZ TEST       LOCATION                  ASSESSMENT
 9.0       focused_porcupine   heap_buffer_overflow  fuzz_exploreMe  src/explore_me.cpp:18:11  n/a
 2.0       suspicious_opossum  shift_exponent        fuzz_exploreMe  src/explore_me.cpp:13:11  n/a

Use 'cifuzz finding <finding name>' for details on the finding.


Logs can be found here:
        /Users/d0i2ceo/ci-fuzz-cli-tutorials/example-projects/simple-setup/cmake/.cifuzz/logs/spark.log
d0i2ceo@CDDEINGMC002239 cmake %
```



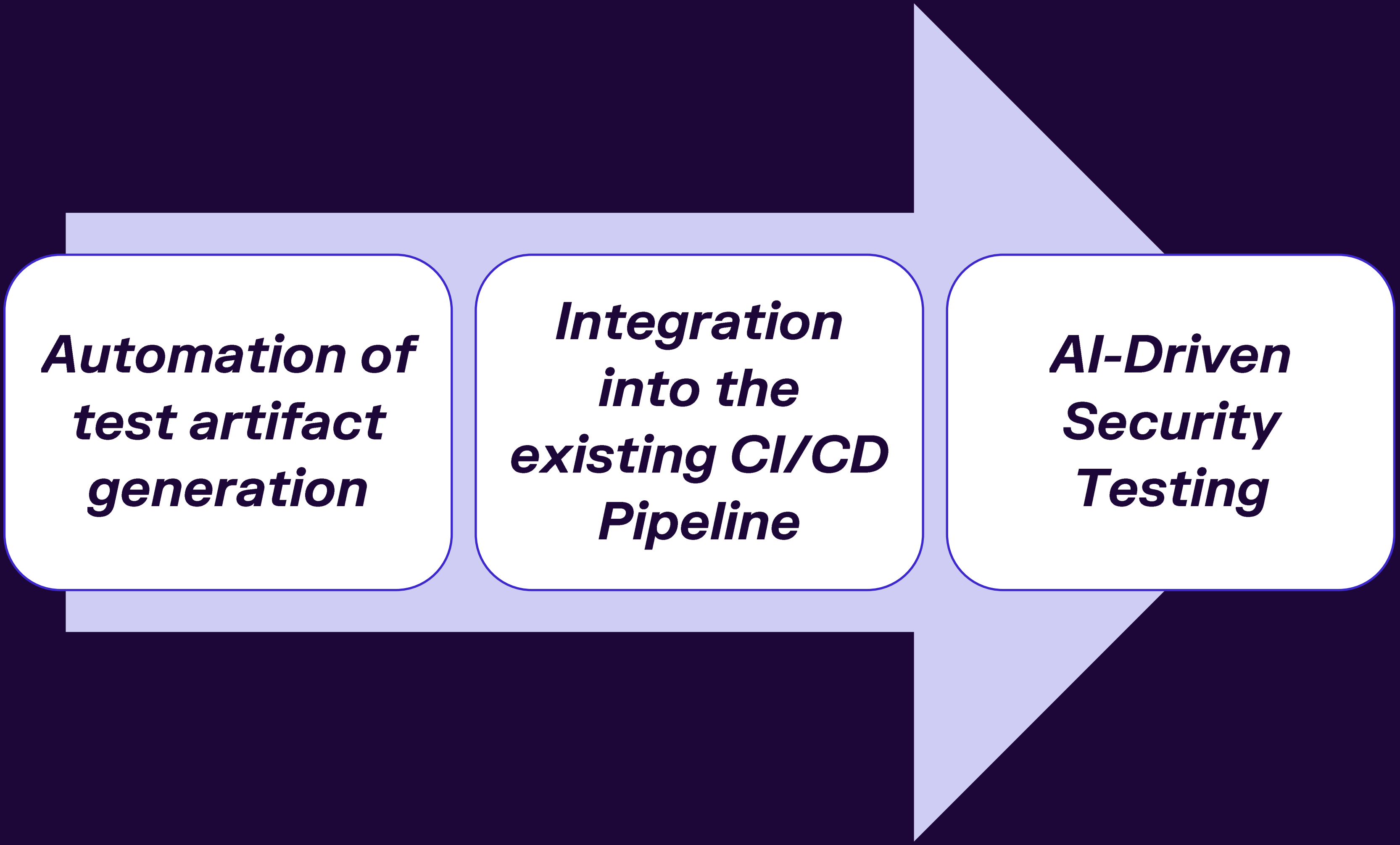
```
ses       103     branches
          31960   inputs
          0       exec/s

            DESCRIPTION         FUZZ TEST             LOCATION
rent_hare   heap_use_after_free test_me_fuzztest      fuzztests/test_me_fuzztest.cpp:10:5
rent_duck   shift_exponent      explore_me_fuzztest   main/src/explore_me.cpp:11:11

<name>' for details on the Finding.
 have been added to the corpus and will now be used to generate more inputs and for automatic
xplore_me_fuzztest/reverent_duck-crash-98b2a3415b62350c11567e6c9dc82ef643d792c4
est_me_fuzztest/reverent_hare-crash-8a91537bcd3a83491ed2d3f4d011f99e7e123a3f

re:
eo/ci-fuzz-cli-tutorials/example-projects/advanced-setup/cmake/.cifuzz/logs/run.log
eo/ci-fuzz-cli-tutorials/example-projects/advanced-setup/cmake/.cifuzz/logs/build-explore_me_
eo/ci-fuzz-cli-tutorials/example-projects/advanced-setup/cmake/.cifuzz/logs/build-test_me_fuz

ct coverage information?
/Users/d0i2ceo/.local/bin/cifuzz coverage --engine=libfuzzer-clang' [Y/n]Y

ocess...
0
re_me_fuzztest...
v report for explore_me_fuzztest...
me_fuzztest...
v report for test_me_fuzztest...
age report...

          File | Functions Hit/Found |  Lines Hit/Found | Branches Hit/Found
e_fuzztest.cpp |       2 / 2 (100.0%) |   6 / 6 (100.0%) |     0 / 0 (100.0%)
ib/test_me.cpp |       1 / 1 (100.0%) |   9 / 9 (100.0%) |     2 / 2 (100.0%)
explore_me.cpp |       1 / 1 (100.0%) | 11 / 15  (73.3%) |     7 / 8  (87.5%)
e_fuzztest.cpp |       2 / 2 (100.0%) |   8 / 8 (100.0%) |     0 / 0 (100.0%)
               |                      |                  |
               | Functions Hit/Found |  Lines Hit/Found | Branches Hit/Found
         Total |       6 / 6 (100.0%) | 34 / 38  (89.5%) |     9 / 10  (90.0%)

re:
eo/ci-fuzz-cli-tutorials/example-projects/advanced-setup/cmake/.cifuzz/logs/coverage.log
239 cmake % 
```



```
cifuzz version 6.15.0
 ✅  Building explore_me_fuzztest...
 ✅  Generating lcov report for explore_me_fuzztest...
 ✅  Building test_me_fuzztest...
 ✅  Generating lcov report for test_me_fuzztest...
 ✅  Creating coverage report...

                          File | Functions Hit/Found |  Lines Hit/Found | Branches Hit/Found
    fuzztests/test_me_fuzztest.cpp |    2 / 2 (100.0%) |   6 / 6 (100.0%) |    0 / 0 (100.0%)
                  lib/test_me.cpp |    1 / 1 (100.0%) |   9 / 9 (100.0%) |    2 / 2 (100.0%)
           main/src/explore_me.cpp |    1 / 1 (100.0%) | 11 / 15  (73.3%) |    7 / 8  (87.5%)
main/tests/explore_me_fuzztest.cpp |    2 / 2 (100.0%) |   8 / 8 (100.0%) |    0 / 0 (100.0%)
                                   |                    |                  |
                                   | Functions Hit/Found |  Lines Hit/Found | Branches Hit/Found
                             Total |    6 / 6 (100.0%) | 34 / 38  (89.5%) |    9 / 10  (90.0%)

Logs can be found here:
        /Users/d0i2ceo/ci-fuzz-cli-tutorials/example-projects/advanced-setup/cmake/.cifuzz/logs/coverage.log
d0i2ceo@CDDEINGMC002239 cmake % cifuzz spark
cifuzz version 6.15.0
 ✅  Validating build system configuration...
 ✅  Configuring CMake project...
 ✅  Analyzing files... 2 candidates found.
 ✅  Calculating coverage of existing fuzz tests...
 ❌  Fuzz test for "exploreMe"... Variant 1/1... Building (Attempt 3)... Error!
 ❌  Fuzz test for "testMe"... Variant 1/1... Running (Attempt 3)... Error!
No more fuzz test candidates.

    FUNCTION   STATUS   LOCATION
 1  exploreMe  Failed   cifuzz-spark/fuzz_exploreMe.cpp
 2  testMe     Failed   cifuzz-spark/fuzz_testMe.cpp

 🚀 0 successful fuzz tests.
 🚀 0 Findings detected.
 🚀 0 Unique Test Cases.
 🚀 89.47% total code coverage.
Used 50k LLM tokens in 6m9s.


Logs can be found here:
        /Users/d0i2ceo/ci-fuzz-cli-tutorials/example-projects/advanced-setup/cmake/.cifuzz/logs/spark.log
```

Expected Outcomes

Expected Outcomes



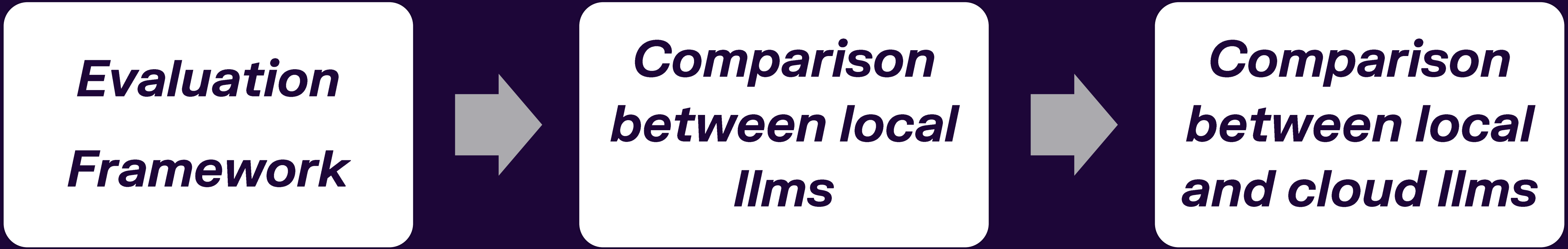
Automation of test artifact generation

Integration into the existing CI/CD Pipeline

AI-Driven Security Testing

Next Steps

Next Steps



Evaluation Framework

Comparison between local llms

Comparison between local and cloud llms

Evaluation Framework

Metrics:

- Number of bugs found
- Code coverage percentage
- Time to generate fuzz tests
- Quality of generated code (measured via static analysis or manual review)
- Number of tokens

Conclusion

This thesis focuses on integrating AI and LLMs into CI/CD/CT pipelines to improve the security testing of automotive software

Thank you!

We transform automotive mobility

CARIAD

A VOLKSWAGEN GROUP COMPANY

Any Questions