**AFL Research**

* Highly regarded **fuzz testing tool** used for finding bugs, security vulnerabilities, and crashes in software.
* Works by generating random test cases (inputs) and feeding them to a program to discover unexpected behaviours, such as crashes or memory corruptions

**Key Features of AFL**

1. **Instrumentation-Based Coverage**: AFL uses code instrumentation to monitor how the program executes with different inputs. This helps it understand what parts of the program are being tested, guiding it to explore untested paths.
2. **Intelligent Input Mutation**: AFL doesn't just generate random inputs; it modifies existing inputs (seed files) intelligently to increase the chances of finding new paths and vulnerabilities.
3. **Crash Discovery/Debugging**: It identifies inputs that cause crashes or unexpected behaviour, helping developers debug and fix issues.
4. **Parallel Fuzzing**: AFL supports running multiple instances in parallel, which speeds up testing on multi-core systems.
5. **Minimization Tools**: It provides utilities to minimize crashing test cases and reduce input files to their smallest interesting forms for easier debugging.
6. **Visual Statistics and Feedback**

* Provides a real-time terminal interface that shows:
  + Number of crashes and hangs found.
  + Progress through different code paths.
  + Mutation strategies being used.
  + Performance metrics like executions per second.

Applications

* **Security Testing**: Identifying vulnerabilities like buffer overflows, memory leaks, or use-after-free bugs.
* **Robustness Testing**: Ensuring software handles edge cases gracefully.
* **Compatibility Testing**: Verifying how software behaves with malformed or unexpected inputs.

Advantages

* User-friendly and requires minimal configuration.
* Efficient at discovering deep and obscure bugs.
* Open-source and widely adopted in the security and software testing communities.

Github link: https://github.com/google/AFL