code intelligence

Breakthrough Dynamic Application Testing

CI Fuzz is a state-of-the-art security testing software. It offers easy IDE integration that saves developers' time and effort while drastically improving the stability and reliability of the codebase.

Why Code Intelligence?

- **Get access to state-of-the-art technology:** Feedback-based fuzzing and concolic code execution
- **Achieve reliable testing results:** Almost no false positives due to the combination of dynamic and static analysis
- **Improve the discovery of vulnerabilities:** Higher code coverage
- **No additional effort:** IDE integration helps to define tests while writing the code
- **Maximize your productivity:** Browse and replay the found bugs and fix them more quickly
- **Test when and where you need it:** Fast and reliable source code testing integrates into your CI/CD process

Technical features

- Supplements feedback-based fuzzing with concolic execution
- Combines several fuzzing engines: AFL++, libFuzzer with Sanitizers and honggfuzz
- Additionally includes classic fuzzing approaches generating patterns such as radamsa
- Utilizes grammar-aware fuzzing for structured inputs
- Uses a framework similar to Qsym and Driller for concolic execution
- Includes, where applicable, APIs and network sockets into testing

Code Intelligence GmbH

Rheinwerkallee 6
53227 Bonn
www.code-intelligence.com

Kontakt

Jonathan Reimer
+49 228 2869 5830
sales@code-intelligence.com

Follow us

@CI_GmbH
company/codeintelligence/

code intelligence

Methods used

- Initial static analysis

  Our software - CI Fuzz - scans the source code to identify which parts can be tested dynamically and guide the mutation engines by providing hints for data structures. If our static analysis identifies external interfaces such as sockets or external API functions, we recommend fuzzing those interfaces.

- Dynamic testing

  The application is tested during its execution. The output derived in the fuzzing process is analyzed and continuously used as input for subsequent test runs. Due to execution testing, virtually all vulnerabilities can be discovered and the number of false positives practically reduced to zero.

- Feedback-based fuzzing

  The software under test is fed with inputs, which are purposefully mutated in the testing process. The fuzzer gets feedback about the code covered during the execution of a given input. This allows the fuzzer to explore the program state efficiently and thus significantly increase the chances of triggering vulnerabilities.

Usability

CI Fuzz offers an easy to use interface to apply these advanced technologies. No deep technical knowledge of fuzzing is required. Instead, users just define which functions or interfaces (e.g. network sockets) they want to have tested and our software does the rest.

Our IDE plugin displays which parts of the code have been reached by the fuzzer and visualizes the fuzzing process. Found crashes can also be replayed by starting the IDE's debugger with the input causing the crash. Alternatively, you can interact with the core software using the command line.

Continuous integration

Code Intelligence software easily integrates into a standard CI / CD workflow such as Jenkins, the fuzz tests are run automatically with each new code change and incidents are reported timely. We also handle special requests for fuzzing on a Kubernetes cluster.

Code Intelligence GmbH

Rheinwerkallee 6
53227 Bonn
www.code-intelligence.com

Kontakt

Jonathan Reimer
+49 228 2869 5830
sales@code-intelligence.com

Follow us

@CI_GmbH
company/codeintelligence/