Fraunhofer FOKUS

Building Trust and Embracing Regulation

Comprehensive Security Testing Made Easy

FUZZINO

Comprehensive Security Testing Made Easy

Our security testing solution Fuzzino employs advanced fuzzing techniques to provide superior test coverage for all your software components. It enables you to mitigate vulnerabilities in your products and reduce liability risks, ensuring compliance with the Cyber Resilience Act, without requiring specialized security testing skills.

The number of companies being attacked is increasing every year, with a new peak of 60 percent in 2024, accumulating a damage of more than 200 billion euros in Germany alone, roughly 4 percent of its GDP. The European Union is adopting regulations, e. g., the Cyber Resilience Act, to counteract this trend by imposing new obligations on manufacturers regarding product security, including regular security testing to identify vulnerabilities. These new obligations are reinforced by the new Product Liability Directive, which makes manufacturers of digital products liable not only for damages caused by vulnerabilities, but also for all damages caused by an attacker exploiting a vulnerability. The new regulations

Main obligations from the Cyber Resilience Act pertaining to Security Testing:

1. Identify and document vulnerabilities, including third-party and open-source software.
2. Promptly address and remediate vulnerabilities, by providing security updates.
3. Apply effective and regular tests and reviews of the security of the product.

apply to many manufacturers who weren‘t previously required by law to perform security testing. In addition, the obligations apply not only to the manufacturer’s source code, but also to any included third-party software component. The explicit inclusion of open-source software presents new technical challenges for manufacturers.

The next level of security testing: identifying vulnerabilities deep in the system

Security testing has heavily evolved in the last 30 years to a large suite of methodologies, techniques, and tools. However, their sheer number can still be overwhelming. Fuzzing is one of the most widely used security testing techniques. Its core idea is to execute a system with randomly generated inputs to uncover vulnerabilities. Even though fuzzing can be quite effective, many fuzzing tools suffer from certain challenges and limitations. These can be significant, particularly to small and medium-sized companies which may lack experience or resources.

Fuzzino is designed with usability in mind: It does not require specific security testing skills, so any tester can benefit

Our solution Fuzzino addresses these challenges and limitations through a set of thoroughly developed features. As a result, it significantly enhances the usability, efficiency, and effectiveness of security testing. These improvements enable manufacturers to leverage state-of-the-art security testing without the need for specialized and expensive security testing training.

The key features of Fuzzino comprise:

- **Usability:** Fuzzing is associated with the need for specific skills to use those tools, how to integrate them in a test environment, and how to interpret their results, which hinders their wide adoption. To provide the output of a fuzzer as input to a component, often a so-called fuzzing harness is necessary. Its implementation requires manual effort and knowledge of the component and of the fuzzing tool. Our solution overcomes fuzzing harnesses, since it enables you to reuse your adapter from functional testing without any changes.
- **Speed:** Today’s fuzzing tools often require long runtimes, i. e., hours or days, to identify vulnerabilities, which often makes them unsuitable for the integration in daily tests, build pipelines, and DevOps processes. Our solution has been developed with speed in mind. Its advanced test

generation engine allows you to identify vulnerabilities in minutes instead of hours.

- **Adaptive vulnerability detection:** Vulnerabilities can manifest themselves in crashes or memory corruptions. However, bugs often have more subtle effects. Existing fuzzers fail to detect these types of vulnerabilities since they focus on memory corruption. Moreover, most fuzzing tools limit themselves to maximizing code coverage, which does not necessarily correlate with vulnerability discovery. Our solution can go beyond maximizing code coverage as a test objective. It can be configured to observe any runtime property, e. g., to identify denial of service vulnerabilities as a test objective.
- **Statefulness**: Many vulnerabilities are hidden deep in the business logic of the system. Many fuzzers can’t find them as their inputs are rejected in early processing stages and as these fuzzers do not operate statefully. Our solution provides a simple and intuitive language to describe message sequences, and thus system states. Operating on these descriptions, our solution effectively finds vulnerabilities deep in the system that other fuzzing tools would miss.

> „
>
> Manufacturers will have to take responsibility for the cybersecurity of their products and applications throughout their entire life cycle.“
>
> **Claudia Plattner,** BSI President

Simply put, fuzzing involves shooting many arrows at a software component under test to find vulnerabilities in it

- **Message descriptions:** Many fuzzers require well-formed example inputs, so-called seeds. However, obtaining a set, which is concise and representative, is not trivial. Our solution employs scalable protocol descriptions. These allow you to describe protocol compliant inputs not only by example inputs (i.e., seeds), but also by describing their data structures and rules they follow. Such descriptions can often be directly obtained or derived from official specifications.

Developers and testers without security testing knowledge benefit from these features to perform comprehensive security testing of any software components within a product.

Two Open-Source Case Studies

The effectiveness of Fuzzino has been confirmed on two open-source case studies: Eclipse® Mosquitto and NanoMQ. Both are brokers for the MQTT protocol, which is often used in the Internet of Things to exchange messages in domains such as manufacturing, automotive, and agriculture. We have subjected these brokers to fuzzed inputs on their functional MQTT interfaces, i.e., without a dedicated adapter for security testing. Fuzzino generated and sent fuzzed MQTT messages to the brokers. These messages also considered the server state, which allowed Fuzzino to produce more complex interactions than traditional fuzzers. As a result, our solution found multiple zero-day vulnerabilities in both brokers.

In Mosquitto, Fuzzino has uncovered a high-severity vulnerability (CVE-2024-8376) whose effects range from resource exhaustion to invalid memory accesses, and which can crash the entire broker. Most notably, we detected this vulnerability in just a few minutes. In contrast, Google OSS fuzz continuously tested this Mosquitto release for more than a year with several open-source fuzzers without producing this vulnerability. On NanoMQ, Fuzzino uncovered two zero-day vulnerabilities which produce invalid memory accesses in a couple of minutes each. This is a drastic improvement compared to the hours or days of fuzzing, which are more common in the industry.

Our security testing solution Fuzzino significantly enhances the efficiency, effectiveness, and usability of security testing. By generating and sending fuzzed messages that consider the server state, it rapidly identifies complex, high-severity vulnerabilities. This approach outperforms traditional fuzzers, reducing detection time from hours or days to mere minutes, and demonstrating superior capabilities in uncovering critical issues.

Contact

Dipl.-Inform. Martin Schneider
Head of Testing
Business Unit Quality Engineering
Phone +49 30 3463-7383
martin.schneider@fokus.fraunhofer.de

Fraunhofer FOKUS
Kaiserin-Augusta-Allee 31
10589 Berlin
Germany

www.fokus.fraunhofer.de/en/sqc/security_testing

www.fokus.fraunhofer.de/en

We
connect
everything

iku | 2502 (Photos: Cover: AI-generated / Fraunhofer FOKUS,
p. 3-4: istock / SeventyFour; p. 6: AI-generated / adobe stock)