Projektgruppe Sicherheit in verteilten Systemen

BA-INF 051

Anmeldung

Um ein Thema zu erhalten, musst du dein Expose bis zum 07.10.2020 eingereicht haben. Sende es bitte an den Betreuer des jeweileigen Themas. Du wirst bis zum 14.10.2020 informiert, ob du dein gewünschtes Thema erhalten/nicht erhalten hast.

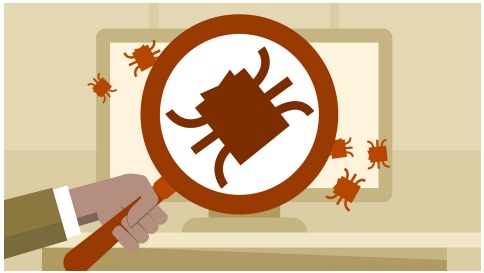
Deadline für das Exposé: 07.10.2020

Deadline für die Registrierung in BASIS: 15.11.2020

Fuzzing

Betreuer

Klaus Tulbure
tulburek@iai.uni-bonn.de
Mischa Meier
meierm@cs.uni-bonn.de



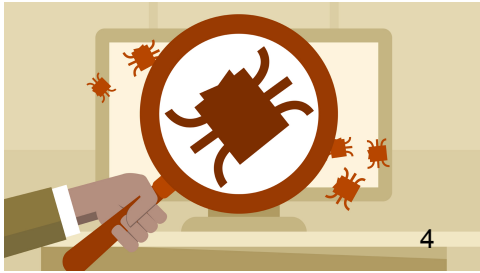
Fuzzing

Fuzzing ist eine der am meisten genutzten Programmanalysen, um Bugs und Sicherheitslücken in Software zu finden. Man kann es einfach gesagt als "intelligentes Brute-Forcing" bezeichnen, das ein Programm solange mit Input bombardiert, bis es hoffentlich irgendwann abstürzt. Als Grundlage nutzen die meisten Fuzzer einen evolutionären Algorithmus und kombinieren darin dutzende verschiedene Algorithmen und eine Vielzahl an Techniken, unter anderem statische Softwareanalysen, Machine Learning, Hardwarebeschleunigung, usw.

Deine Aufagbe ist es, einen Fuzzer zu erweitern. Du kannst dabei selbst entscheiden, ob du eine bestehende Technik verbessern willst, sie auf einen anderen Fuzzer übertragen oder deine eigene entwickeln möchtest. Du kannst eigene Ideen vorschlagen oder bekommst eine von uns.

Vorraussetzung sind gute Programmierkenntnisse in C/C++ oder Java.

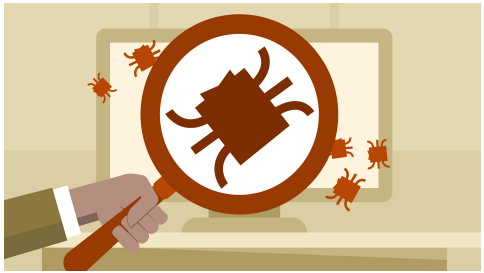
Literatur https://www.cs.umd.edu/~mwh/papers/fuzzeval.pdf



Anti- and Anti-Anti-Fuzzing-Techniken

Supervisor

Klaus Tulbure
tulburek@iai.uni-bonn.de



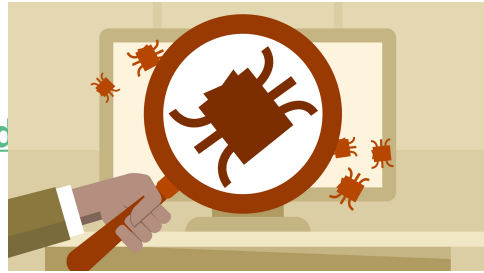
Anti- and Anti-Anti-Fuzzing-Techniken

Einige Entwickler wollen nicht, dass ihre Software effizient gefuzzt werden kann, um sich zum Beispiel vor Angreifern zu schützen. Aus diesem Grund implementieren sie **Anti-Fuzzing-Techniken**, die den Fuzzing-Prozess verlangsamen. Allerdings ist dies nur eine Scheinsicherheit (also nicht gut) und kann mittelfristig von einem Angreifer leicht überwunden werden. Daher gilt es, diese Anti-Fuzzing-Techniken zu knacken und die Sicherheitslücken zu schließen, bevor böswillige Angreifer dies tun.

Deine Aufgabe ist es, eine solche Anti-Fuzzing-Technik zu entwickeln,
oder, einen Fuzzer so zu erweitern, dass er eine bestehende Anti-Fuzzing-Technik schlägt.

Vorraussetzung sind gute Programmierkenntnisse in C/C++ oder Java.

Literatur https://www.usenix.org/system/files/sec19fall_jung_prepub.pd

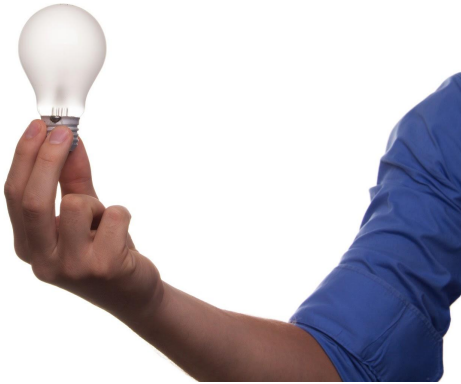


Eigene Ideen?

Betreuer

jeder von uns

Email an
Klaus Tulbure
tulburek@iai.uni-bonn.de



Eigene Ideen?

Du hast ein paar eigene Ideen für ein Projekt im Bereich Usable Security and Privacy?

GROßARTIG!

Wir würden uns freuen, mit euch zusammenzuarbeiten!