FG Software Engineering
Leibniz Universität Hannover
Welfengarten 1
3. Stock, Flur G

# Master-Arbeit

**- zu vergeben -**

## Security Runtime Monitor

### *Hintergrund*

Security Laufzeit Analysen werden aufgrund von ständig ändernden Umgebungsbedingungen immer wichtiger. Sobald sich beispielsweise das System ändert auf welchem die Software ausgeführt wird oder neue Technologien entstehen, welche zum Entwurfszeitpunkt der Software nicht bekannt waren, könnte dies einen Affekt auf die Security von Software haben.

Die Ausführung eines Programmes kann durch die Reihenfolge in welcher Methoden ausgeführt werden (Sequenzen) beschrieben werden. Diese Methodensequenzen können in Form von Modellen wie z.B. Abstrakt Syntax Trees dargestellt und verwaltet werden. Somit können zwei Methodenausführungen auf Basis dieser ebenfalls miteinander verglichen werden um Unterschiede derer oder gar Abweichungen zu regulär enthaltenen Sequenzstrukturen festzustellen. Um Software Executiontraces zu erzeugen gibt es bereits Frameworks die das Erstellen von Traces unterstützen (Kieker[1]).
[1] - http://kieker-monitoring.net/framework/

### *Aufgabe*

Im Rahmen dieser Arbeit soll ein Java Anwendung entwickelt werden, die ein/e Javaprojekt/Jar Datei als Eingabe erhält und ein Modell erstellt, welches die möglichen Methodensequenzen der Anwendung beschreibt. Weiterhin soll die Anwendung dann ausgeführt und getract werden, wobei anhand der mitgeloggten Tracedateien unterschiede der Methodensequenzen in der Projektstruktur enthaltenen Methodenlogik erkannt werden sollen. Dies soll durch die Differenzbildung von Modellen erfolgen.

Sollten unterschiede erkannt worden seien, ist zu prüfen ob die Ausführungsreihenfolge des Befunds der Methodensequenz von Exploit als auch Schwachstellenbehafteten Code, der in einem Repository abgelegt ist, ähnelt. Bestehende anwendungsformen von Modelldifferenzbildung müssen auf Basis der Effizienz analysiert und verglichen werden um bestmögliche Ergebnisse zu erzielen (Responsetime etc.). Weiterhin muss sich ein Konzept der Darstellung überlegt werden, welches diese Befunde dem Entwickler, in Form eines Berichts, mitteilt. Die Analyse des Programmablaufs soll fortlaufend während der Ausführung eines Programmes durchgeführt werden und darf die Ausführung dieses Programmes nicht beeinflussen (Keine Ladezeiten/Stopper).

Anforderungen:
- Literatursuche: Ansätze Modelldifferenzierung & Darstellung von Source Code als Modell
- Analyse bestehender Modelldifferenzierungsansätze
- Gut strukturierter und kommentierter Programmcode (Modular gekapselt)
- Entwicklung einer Java-Anwendung
- Evaluation des erstellten Ansatzes (Precision, Recall und Response Time)

### *Organisatorisches*

**Betreuer**: M. Sc. Fabien P. Viertel, fabien.viertel@inf .uni-hannover.de, Raum G307
**Prüfer:** Prof. Dr. Schneider **Beginn**: Ab sofort möglich