Softwareprojekt: Iridium

ein Analyseframework für kompilierte Programme

Inokentiy Babushkin

Abteigymnasium Brauweiler

11.05.2015 / Colloquium zur Besonderen Lernleistung

Inhalt

Grundlagen

- Kurze Einführung in CPU-Interna
- Reverse Engineering

Features

- Prinzipien
- Kontrollflussanlyse
- Analyse von Daten auf dem Stack
- Analyse optimisierter Integerdivisionen

Inhalt

Grundlagen
- Kurze Einführung in CPU-Interna
- Reverse Engineering

Features
- Prinzipien
- Kontrollflussanlyse
- Analyse von Daten auf dem Stack
- Analyse optimisierter Integerdivisionen

- Die CPU unterstützt einzelne Instruktionen, darunter auch Sprünge
- Diese werden durch sog. Opcodes im Speicher dargestellt
- Können auch in Form von Assembler-Befehlen beschrieben werden
- Vom Abstraktionsgrad her kaum mit Hochsprachen zu vergleichen

Inhalt

Grundlagen
- Kurze Einführung in CPU-Interna
- Reverse Engineering

Features
- Prinzipien
- Kontrollflussanlyse
- Analyse von Daten auf dem Stack
- Analyse optimisierter Integerdivisionen

Definition hier: Wiederherstellung von Hochsprachencode aus kompilierten Programmen

- Komplexer und zeitaufändiger Prozess, da
  - Software zunehmend komplex
  - Compiler den Code optimieren
- Großer Nutzen für verschiedene Bereiche, z.B:
  - Malware-Analyse
  - Wiederherstellung von altem Programmcode
  - Entwicklung von Exploits

(Halb-)Automatisierte Ansätze für die statische Analyse

- Decompiler
- Analyseframeworks
- Disassembler

Inhalt

Grundlagen
- Kurze Einführung in CPU-Interna
- Reverse Engineering

Features
- Prinzipien
- Kontrollflussanlyse
- Analyse von Daten auf dem Stack
- Analyse optimisierter Integerdivisionen

Erweiterbarkeit

- Modularer Aufbau
- Sinnvolles API-Design
- Multiple Eingabeformate
- Lesbarkeitsorientierter Code

Freie Software

- Verwendung der GPLv3
- Codequalität und Funktionsumfang kann durch die Community erweitert werden

Inhalt

Grundlagen
- Kurze Einführung in CPU-Interna
- Reverse Engineering

Features
- Prinzipien
- Kontrollflussanlyse
- Analyse von Daten auf dem Stack
- Analyse optimisierter Integerdivisionen

Analyse anhand von Kontrollflussgraphen

- Schrittbasierter Reduktionsalgorithmus für einzelne Funktionen
- Unterscheidung aller wichtigen Hochsprachen-Programmelemente
- Strukturierte Darstellung

Inhalt

Grundlagen
- Kurze Einführung in CPU-Interna
- Reverse Engineering

Features
- Prinzipien
- Kontrollflussanlyse
- Analyse von Daten auf dem Stack
- Analyse optimisierter Integerdivisionen

Heuristiken zur Untersuchung lokaler Variablen

- Untersuchung des Stackframes der jeweiligen Funktion
- (Teilweise) Erkennung von Arrays, Datentypen und Pointern

Inhalt

Grundlagen
- Kurze Einführung in CPU-Interna
- Reverse Engineering

Features
- Prinzipien
- Kontrollflussanlyse
- Analyse von Daten auf dem Stack
- Analyse optimisierter Integerdivisionen

Optimisierte Integerdivision

- Von modernen Compiler bei konstantem Divisor angewendet
- Zeitintensive Divisions-Instruktion wird durch eine Reihe von Additionen, Shifts und Multiplikationen ersetzt
- Für Menschen keine Rückschlüsse auf den Divisor möglich, deswegen Implementation eines entsprechenden Algorithmus

Zusammenfassung

- Reverse-Engineering und Analyse kompilierter Programme sind häufig notwendig, allerdings meist auch schwieig und mit enormem Aufwand verbunden.
- Gleichzeitig sind die automatisierten Lösungsansätze für diese Aufgaben nicht oder nur kaum für interaktive Arbeit geeignet, weswegen mit dem vorliegenden Projekt ein entsprechender Prototyp vorgelegt wird.

- Ausblick
  - Stabilisierung des Codes
  - Erweiterung um Eingabemodule
  - Weitere Features