software visualization

NICK MÜLLER, a1001030

Papers

- Visualizing Compiled Executables for Malware Analysis. Daniel A. Quist and Lorie M. Liebrock, VizSec 2009.
- Visualizing Application Behavior on Superscalar Processors. Chris Stolte, Robert Bosch, Pat Hanrahan, and Mendel Rosenblum, Proc. InfoVis 1999

Zusammenfassung –

Visualizing Compiled Executables for Malware Analysis

- Reverse Engineering von Software teilweise sehr aufwändig und schwer
- Analyse von Malware durch „Armor“ weiter erschwert
- Grundlegende Erklärung von Reverse Engineering
- Einführung eines Frameworks, um Analyse von Software zu erleichtern

Reverse Engineering – Überblick

- Analyse von Software ohne vorliegenden Sourcecode
- Oft direkte Interaktion mit Assembler-Code
- Untersuchung mit verschiedenen Tools/Methoden möglich:
  - Statische Tools
  - Dynamische Tools
- Analyse kann durch Obfuscation/Armoring verhindert werden

Reverse Engineering – Workflow

- Erstellung einer isolierten Umgebung
- Ausführung des Programmes und Überwachung von Änderungen
- Untersuchung des Programmes mit Reverse Engineering Tools
  - ggf. Deobfuscation / Entfernung von Code-Schutzmaßnahmen
- Identifizierung von relevanten Programmteilen zur genaueren Untersuchung

VERA Architecture

- Aufteilung des Tools in drei Teile:
  - Hypervisor-based Monitoring Framework
  - System zur Erstellung von gewichteten Graphen
  - Darstellung und Interaktion mit Graphen
- Direkte Interaktion mit IDA Pro
- Bietet schnellen Überblick über Struktur eines Programms
- Erleichtert Identifizierung von relevanten Codebereichen

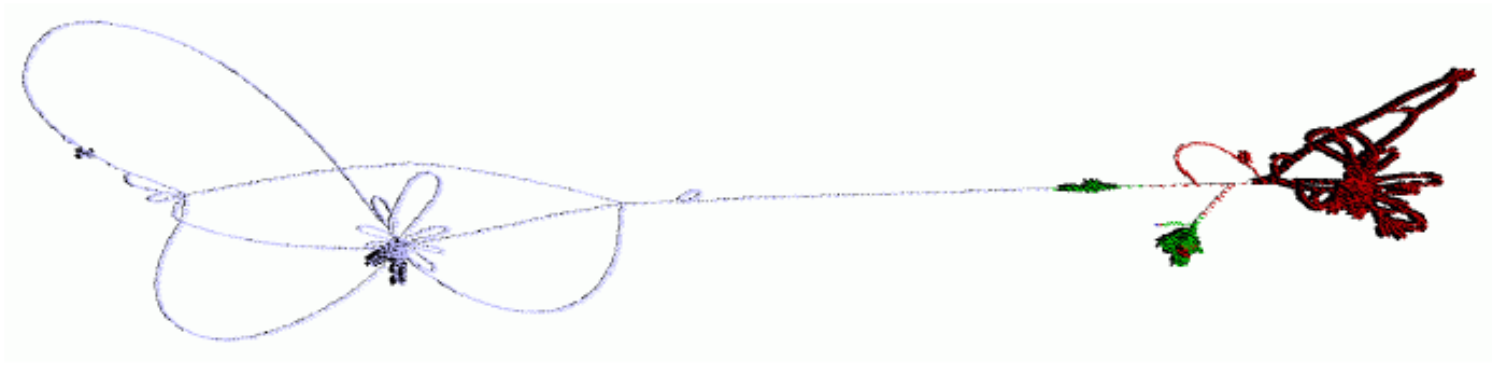


Figure 1: Visualization of the Netbull Virus Protected with the Mew Packer

Figure 6: Mebroot overview of entire execution process

Zusammenfassung –

Visualizing Application Behavior on Superscalar Processors

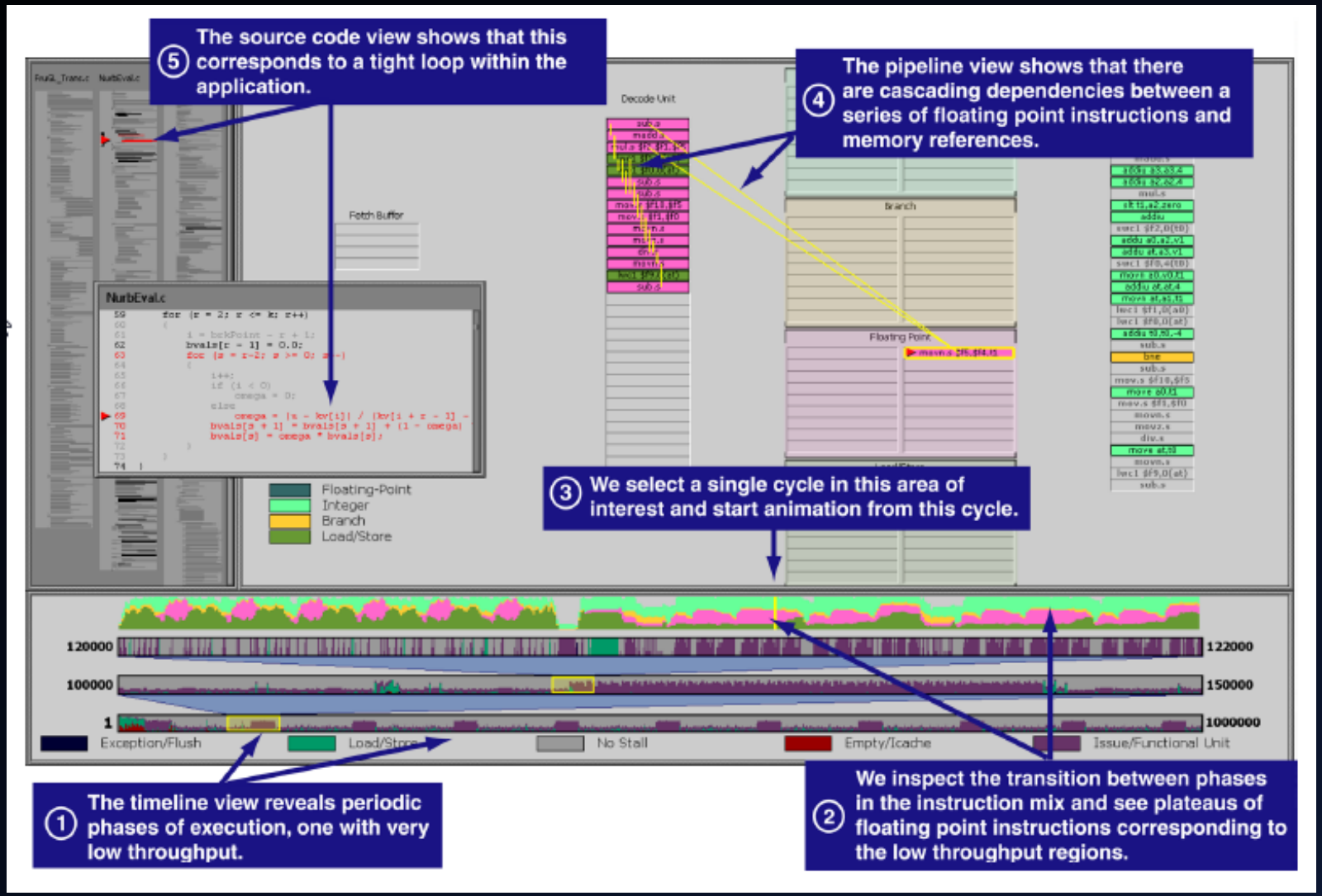
- Superskalare Prozessoren machen Einschätzung von Skalierung / Ausnutzung schwer
- Optimierung im Prozessor für Programmierer meist „unsichtbar“
- Auffinden von Performanceprobleme durch Highlevel-Analysen mühsam
- Einführung eines Frameworks mit overview-plus-detail Display für Anwendungs-Ausführung

Performanceoptimierung

- Unterschiedliche Techniken zur Optimierung:
  - Pipelining
  - Multiple Functional Units
  - Out-of-Order Execution
  - Speculation
- Trotz Optimierung keine optimale Performance möglich
- Leistungsverlust durch Abhängigkeiten oder schlechte Ausnutzung der Pipeline

Performancevisualisierung

- Strukturierung des Systems in drei Kategorien:
  - Timeline View – Probleme finden
  - Pipeline View – Probleme identifizieren
  - Source Code View – Herstellung von Kontext zum Code
- Ermöglicht allgemeine Übersicht über Performance und genaue Analyse von Problemen
- Erlaubt interaktive Untersuchung von Ressourcenausnutzung


5 The source code view shows that this corresponds to a tight loop within the application.
4 The pipeline view shows that there are cascading dependencies between a series of floating point instructions and memory references.
3 We select a single cycle in this area of interest and start animation from this cycle.
1 The timeline view reveals periodic phases of execution, one with very low throughput.
2 We inspect the transition between phases in the instruction mix and see plateaus of floating point instructions corresponding to the low throughput regions.
Decode Unit
Fetch Buffer
Branch
Floating Point
NurbEval.c
Floating-Point
Integer
Branch
Load/Store
120000
100000
1
122000
150000
1000000
Exception/Flush
Load/Store
No Stall
Empty/Icache
Issue/Functional Unit

Einsatzgebiete des Frameworks

- Primär für Untersuchung von Performance auf superskalaren Prozessoren entwickelt
- Kann auch in anderen Anwendungsbereichen eingesetzt werden:
  - Compiler Design
  - Hardware Design
  - Simulator Development

Bewertung –

Visualizing Compiled Executables for Malware Analysis

- Allgemeine Beschreibung von Reverse Engineering zu Beginn
- Erklärungen auch für „Laien“ in der Softwareanalyse verständlich
- Erläuterung des erstellten Frameworks mit Anwendungsbeispiel und User Study
- VERA – Interaktion mit IDA Pro, detaillierter Aufschlüsselung von Codeblöcken und Benutzerfreundlichkeit

Bewertung –

Visualizing Application Behavior on Superscalar Processors

- Grobe Erläuterung zu Optimierung in Prozessoren
- Detaillierte Beschreibung der drei Teile des Frameworks mit Screenshots samt Erklärung
- Kurzes Anwendungsbeispiel
- Nennung möglicher weiterer Anwendungsgebiete
- Keine Nennung der eigentlichen Software / des Frameworks (immer nur „our visualization system“)