Introduction

The BitBlaze Platform

Outline

1 Introduction
2 The BitBlaze Platform

What is BitBlaze?

BitBlaze is a binary analysis tool used to aid in reverse engineering by analyzing binary code.

BitBlaze is primarily intended for analyzing crash data to find vulnerabilities.

Basically, it’s used to find out why an application crashed to see if the application can be fixed.

What’s the problem?

The difficulty/laboriousness of sifting through the binary data in crash dumps.

Searching through crashes is a time-consuming task that requires lots of people all looking for the root cause of an issue. The number of crashes a single program can generate after only a little bit of, say, fuzzing is staggering.
In 2009, Ben Nagy gave a talk called "Finding Microsoft Vulnerabilities by Fuzzing Binary Files with Ruby - A New Fuzzing Framework". In the talk he claimed to have found 200,000 crashes by fuzzing Microsoft Word. These 200,000 could be divided into 61 unique types (or bins)[1], of which 4 were serious security vulnerabilities. He was only able to find out which crashes represented vulnerabilities by submitting the crashes to Microsoft so that they could help him analyze them all.

Who Would Use it and Why?

- **Security Researchers** - Have to find the important crashes
- **Developers** - Trying to find vulnerabilities in order to fix them
- **Security Officials** - Searching for hidden code or malware (since so few malware authors release their source code)

Outline

1. Introduction
   - What is BitBlaze?
   - The BitBlaze Platform

Basic Structure

BitBlaze has three major sets of components to handle:

- **Static Analysis** - VINE
- **Dynamy Analysis** - TEMU
- **Symbolic Exploration** - Rudder/BitFuzz/FuzzBALL
Taint Propagation and Spreading

Application taints data (flags), and within TEMU we can trace how the tainted data is propagated through the system.

Tools and Utilities - 1

- trace reader: Displays information about a trace
- x86 slicer: Displays how data propagated through the program
- align: Highlights the differences between two traces, usually one good and one bad
- tracedump: Displays taint info from around registers acting as pointers
- alloc reader: Displays which allocation corresponds to a given address

Resources

  - Crash Analysis Using BitBlaze. Charlier Miller, Juan Caballero, Noah M. Johnson, Min Gyung Kang, Stephen McCamant, Pongsin Poosankam, Dawn Song

Resources cont