Katana: A Userland Toolchain-Oriented Hotpatching System

James Oakley

24 May 2010

Contents

1 Introduction

2 What Katana Does

3 What Katan Does Not Do (Yet)

4 How to Use Katana

Katana is intended to be used in two stages. The first stage generates a patch object from two different versions of an treee. By an object tree, we mean the set of object files (.o files) and the executable binary they comprise. Katana works completely at the object level, so the source code itself is not strictly required, although all objects must be compiled with debugging information. This step may be done by the software vendor. In the second stage, the patch is applied to a running process. The original source trees are not necessary during patch application, as the patch object contains all information necessary to patch the in-memory process at the object level. It is also possible to view the contents of a patch object in a human-readable way for the purposes of sanity-checking, determining what changes the patch makes, etc.

4.1 Preparing a Package for Patching Support

Katana aims to be much less invasive than other hot-patching system and require minimal work to be used with any project. It does, however, have some requirements.
Required CFLAGS:

- -g

Recommended CFLAGS:

- -ffunction-sections
- -fdata-sections

Recommended LDFLAGS:

- –emit-relocs