

Linux Kernel Code Verification

Abhishek Agarkar 07305053

Ajesh Kumar S. 07305065

Annervaz K M 07305063

December 2, 2007

1 Introduction

This project mainly aimed at proving/disproving the desirable properties of some part of Linux kernel code. We identified some parts of the code, and thier properties. We were able to prove some of these properties, formally. We used the BLAST tool for the formal verification part.

2 Experience

We found it difficult to find the right balanced properties between interesting, usefulness and provability. Fiddling around with BLAST was a nice experience, although it gave some bitter experiences when we tried to run on huge code fragments directly from the kernel source.

3 Description

We mainly selected two code snippets from Linux kernel code, from the files

1. `drivers/usb/core/devio.c`
2. `mm/shmem.c`

For feeding the code to BLAST, we extracted the relevant code parts from the source and all dependant parts of the code to make a separate file. This was because BLAST was not working properly with the header files.

3.1 devio.c

The code taken from devio.c was mainly the function

```
static long usbdev_read(struct file *file, char *buf, unsigned int nbytes,
    long *ppos), and the functions that gets called from this. If we assume a precondition 'nbytes > 0' for this function, we can prove that nbytes will remain positive in the post condition. This property is provable only if we enable -clp option with BLAST, else BLAST is not able to prove it.
```

3.2 shmem.c

The code taken from shmem.c was mainly from the function.

```
static void shmem_truncate_range(struct inode *inode, loff_t start,
    loff_t end). This is a function that truncates free memory, at a particular point and range. We tried to fiddle with 2 properties in this function.
```

1. `page_private(subdir) == freed` is a property that must be met at a point in that function, for normal behaviour. We tried to assert this at that point, but were unable to verify (to prove safe) this using BLAST.
2. This is based on a bug fixed on linux kernel. `limit >= SHMEM_NR_DIRECT` is a desirable property, at a particular point in that function, else it would lead to an abnormal behaviour. This was not ensured in earlier versions of the kernel, and was fixed in later versions. The fix was very obvious based on an if condition. We verified it, but as the fix was obvious, the verification was very straight forward.

This function `shmem_truncate_range` has an intricate structure. We tried to identify some properties and very well played around with that code on BLAST, trying with various properties.

4 BLAST

At first we intended to take the file as it was without modifying any of the code in it, but this proved a bit difficult task as the number of header files was large. We found it tough to run blast with all the headers file within, so we went for work around with removing all the header files from the code and put all the definitions of the function that were affecting the property we were intended to prove and the definitions of the structures used. The task was very time consuming as many structures were a set of structures and we had to get the structure definition to a large depth.

Another very unusual problem we faced was after giving the tool with C file, BLAST was showing the results that were unexpected for us i.e. it was not proving the property we wanted to prove. So we tried BLAST with different options provided with it. We were able to prove the property in `usbdev_read` with `-clp` option with no change in the C file provided to it.