Reimplementation of Carburizer

Louisa Bessad

24 juillet 2014

1. Abstract

Carburizer was created to ensure input checking from the driver, failure detection before corruption appears and a fix of device failures. In this way the reliability of systems can be improved. But it does not handle all issues and few platforms or drivers had been tested. The goal of this document is to reimplement some functions of Carburizer with Coccinelle. To do this ; we have to look for different types of code.

2. Looking for different types of code

First, we will look for loop without timeout. We will consider as a timeout a variable that is incremented or decremented in the condition of the loop or another assignment (such as $\ll=$ *or* $\gg=$).

3. Solutions

To find loops without a timeout we will apply two semantic patches to the Linux3.2.59 Kernel. There is a smeantic patch for each type of loop : for and while. The patch will furthermore modify a loop that matches loop which matches to avoid infinite looping. The inserted code is the same as that used by Carburizer. For instance these codes :

```
1  +       unsigned long long delta = (cpu / khz / HZ) * 2;
2  +       unsigned long long _start = 0;
3  +       unsigned long long _cur = 0;
4  +       unsigned long long timeout;
5  +       timeout = rdstcll(start) + delta;
6
7          for (m = vp->mcast_list; m; m = m -> next) {
8                  if (!memcmp(m->addr, addr, ETH_ALEN))
9                          return m;
10
11 +                        if (_cur < timeout) {
12 +                                rdstcll(_cur);
13 +                        }
14 +                        else {
15 +                                break;
16 +                        }
17
18         }
```

Listing 1 – An instance of a possible infinite for loop, the Sun Ethernet driver can hang if the pointer m-¿next has a bad assignment

```
1  +        unsigned long long delta = (cpu / khz / HZ) * 2;
2  +        unsigned long long _start = 0;
3  +        unsigned long long _cur = 0;
4  +        unsigned long long timeout;
5  +        timeout = rdstcll(start) + delta;
6           while ((urb = bfusb_get_completed(data)))
7                   usb_free_urb(urb);
8  +                        if (_cur < timeout) {
9  +                                rdstcll(_cur);
10 +                        }
11 +                        else {
12 +                                break;
13 +                        }
14
15          }
```

Listing 2 – An instance of a possible infinite while loop,The Bluetooth driver can hang if the function never returns 0

4. Results

We have counted the number of each type of loop on the Linux3.2.59 Kernel. Then we have counted the loops found by our patch. During the execution of patches some files have been skipped by Coccinelle to avoid a very long execution time. It appeared that whatever the computer used to execute the patches without the timeout option of Coccinelle, the execution could never end. For those files the analysis must be made by a programer. The following array shows the result of these executions :

	infinite loop	EXN files	total loop
while	3463	82	9342
for	367	61	27616