Fuzzing test

Referent

Google

https://github.com/google/syzkaller

Introduction

Kernel version upgrades hurt

- Functionality-wise
- Performance-wise

We upgraded in SLE12-SP1-SP2

- About 40k LOCs changed)
- Manual inspection impossible

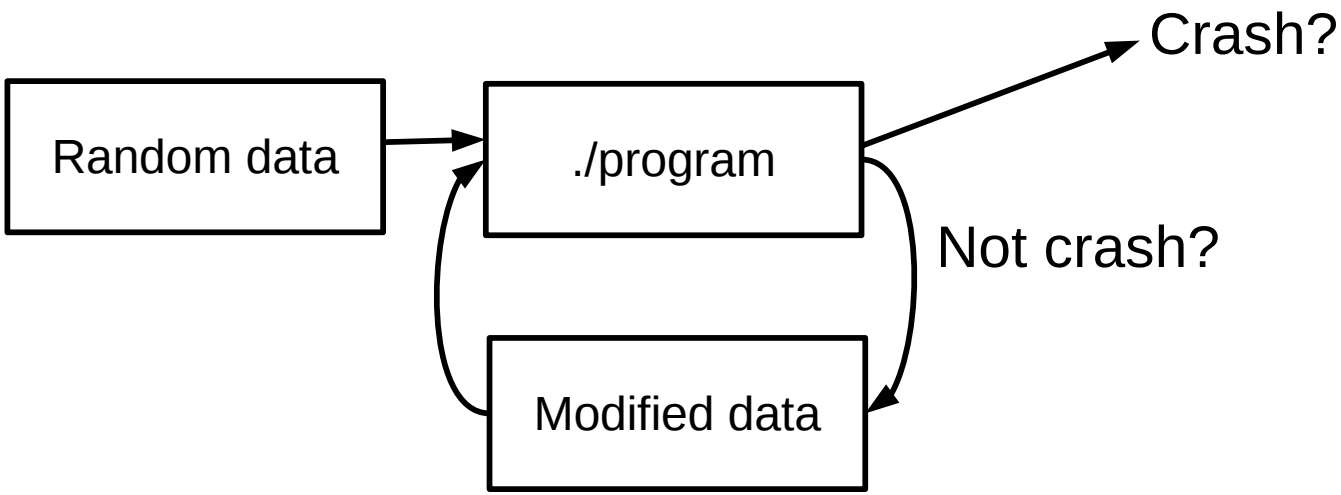
Tools

Not covered: SPARSE, Coverity PREVENT, LDV, . . .

Covered: SYZKALLER,TRINIT

Fuzzing test

Common technique nowadays



Fuzzing the kernel

Userspace fuzzing

- Input from user
- Files from filesystem
- . . .

What is the input of the kernel?

- Data from drivers
  - Network traffic, disk data, . . .
  - These fuzzers do not cover this
- System calls

Trinity

Syzkaller

Syzkaller

Similar to TRINITY

- List of syscalls
- Annotated parameters of every syscall

But it tracks coverage

- It tries to cover all code paths

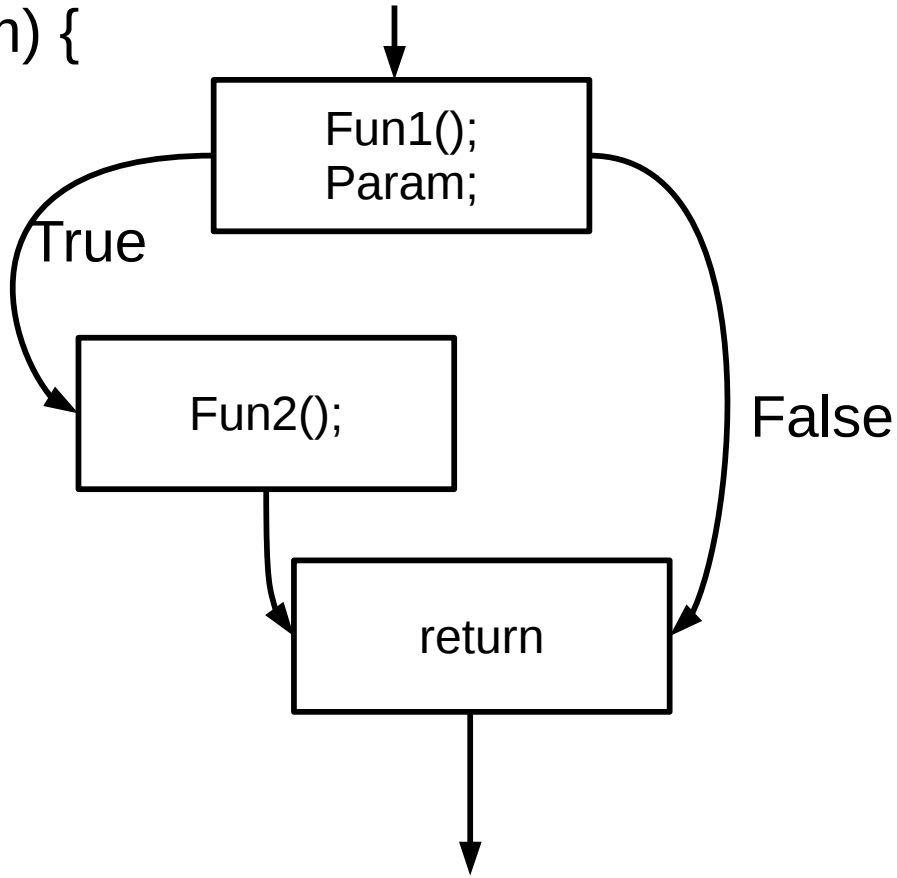
Tracking Coverage

```
1    void syscall_158(int param) {

3      fun1() ;
4      if(param) {

6        fun2() ;
7      }

9    return;
10 }
```



Tracking Coverage

```
1    void syscall_158(int param) {
2      __sanitizer_cov_trace_pc();//  instrumented
3      fun1() ;
4      if(param) {
5        __sanitizer_cov_trace_pc();  //  instrumented
6        fun2() ;
7      }
8     __sanitizer_cov_trace_pc();//  instrumented
9     return;
10  }
```

__sanitizer_cov_trace_pc()

What does it do?

- Stores its caller address
- All exported in debugfs

Who does instrument it into the kernel?

- GCC >=6
- But GCC in SP2 is 4.8 and 5 only

How to enable the instrumentation?

- Turn on CONFIG_KCOV

Requirements

GCC>=6

- Instrumentation support

Kernel

- Enable instrumentation:KCOV
- Enable debugging (optional):KASAN,KMEMLEAK,LOCKDEP
- Disable unwinder (too slow to boot):STACK_UNWIND

SYZKALLER package

QEMU

- SYZKALLER fuzzes in isolation
- Runs in snapshot mode

Description

KASAN – checks memory accesses and helps to catch errors like use-after-free and out-of-bounds. Prior to 4.6,KASAN needs SLUB memory allocator to work correctly (SLES12-SP2's default is SLAB)

KMEMLEAK – reports kernel memory leaks.

LOCKDEP – reports potential deadlocks,stalls,and similar.

CONFIG_KCOV – to store a basic block was visited. The information is then exported via debugfs in /sys/kernel/debug/kcov.

(KCOV is upstreamed in linux 4.6. For older kernels you need to backport

commit : 5c9a8750a6409c63a0f01d51a9024861022f6593.)

Main components of syzkaller

syz-executor – a small,self-contained program containing a sequence of syscalls with input. It is run inside the virtual machine in a hope to cause some harm.

syz-fuzzer – establishes the sequence of system calls,permutes input,compiles all that into several syz-executors. Syz-fuzzer then runs and terminates them. There is a single instance in each virtual machines.

syz-manager – runs and manages virtual machines with syz-fuzzer running inside them. It can also run several virtual machines in parallel. This component is what users are supposed to run.

Principle of Operation

1 syz-manager starts a virtual machine

2 syz-manager starts syz-fuzzer inside the VM

3 Generate a syscalls sequence – syz-executor

- Incl. random data

4 syz-executor is invoked with appropriate parameters

5 Check dmesg for a WARNING/BUG

- Generate a report if there is one

6 Look at the visited paths and permute the input accordingly

7 Check if the VM is viable

- Restart if needed

8 Repeat from 3

Reproduce

SYZKALLER can generate reproducers from reports

- A simple C code
- Triggers the reported WARNING/BUG

Run syz-repro

- Give the report as a parameter
- The reproducer is emitted

syz-repro

- may fail for rare race conditions etc.

How to – host

zypper ar -f http://download.nue.suse.com/ibs/home:/jirislaby:/syzkaller:/sle12-sp2/syz_SLE12-SP2/

zypper in syzkaller

zypper in kvm_server

zypper in -t pattern kvm_server

home:jirislaby:syzkaller contains syzkaller and a new GCC

home:jirislaby:syzkaller:sles12-sp2 always builds the latest patched SLES12-SP2 kernel. The kernel is with all SLAB,KASAN,KMEMLEAK,LOCKDEP, and KCOV enabled.STACK_UNWIND is disabled.

How to – guest

zypper ar -f syz

zypper dup --from syz

zypper in kernel-default-debuginfo

cp vmlinux-<version>,vmlinux-<version>.debug in same directory

vmlinux-<version> can be copied from virtual machine directly(/boot/vmlinux-*.gz) and extracted

zypper in syzkaller

su some_user

allow root logins into virtual machines without asking for password.

ssh-keygen -f my_key

add my_key.pub to /root/.ssh/authorized_keys in VM

How to – config

cp /usr/share/doc/package/syzkaller/example.cfg

```
{
        "http": "localhost:56741",
        "workdir": "/home/kathy/syzkaller/workdir",
        "kernel": "/home/kathy/syz/vmlinuz-4.4.21-56-default",
        "initrd": "/home/kathy/syz/initrd-4.4.21-56-default",
        "vmlinux": "/home/kathy/syz/vmlinux-4.4.21-56-default",
        "image": "/var/lib/libvirt/images/sles-12-sp2-64-fv-def-net.qcow2",
        "sshkey": "/home/kathy/.ssh/id_rsa",
        "cmdline": "root=UUID=42383662-70ca-44c7-83e7-10559a9ddae1",
        "syzkaller": "/usr/",
        "type": "qemu",
        "count": 1,
        "procs": 4,
        "cpu": 2,
        "mem": 2048,
        "disable_syscalls": [
                "keyctl",
                "add_key",
                "request_key"
        ],
        "suppressions": [
                "some known bug"
        ]
}
```

- syz-manager -config example.cfg

Referent

Google

https://github.com/google/syzkaller

Corporate Headquarters
Maxfeldstrasse 5
90409 Nuremberg
Germany

+49 911 740 53 0 (Worldwide)
www.suse.com

Join us on:
www.opensuse.org