Overview of Google **syzkaller**

Jingtang Zhang

2019.07

About **syzkaller**

- Google Security - dynamic tools team
- Golang
- Coverage-guided grammar-based kernel fuzzer

google / syzkaller

Watch 169 | Unstar 2,387 | Fork 512

Code | Issues 101 | Pull requests 16 | Security | Insights

syzkaller is an unsupervised, coverage-guided kernel fuzzer

linux | kernel | fuzz-testing | fuzzing | fuzzer | testing | security | security-vulnerability | security-tools

Go 97.3% | Objective-C 1.9% | C 0.5% | C++ 0.1% | Shell 0.1% | Python 0.1%

Kernel Fuzzers before **syzkaller**

Trinity

```
while (true) {
    syscall(rand(), rand(), rand());
}
```

✓ Argument type

```
while (true) {
    syscall(rand(), rand_fd(), rand_addr());
}
```

- Shallow bugs
- No reproducers

Operation of a Typical Kernel Fuzzer

- **Manually** create a bunch of VMs
- **Manually** copy and start the binary
- **Manually** monitor console output
- **Manually** deduplicate crashes
- **Manually** localize and reproduce
- **Manually** restart the crashed VM / Press power button

Efficiency

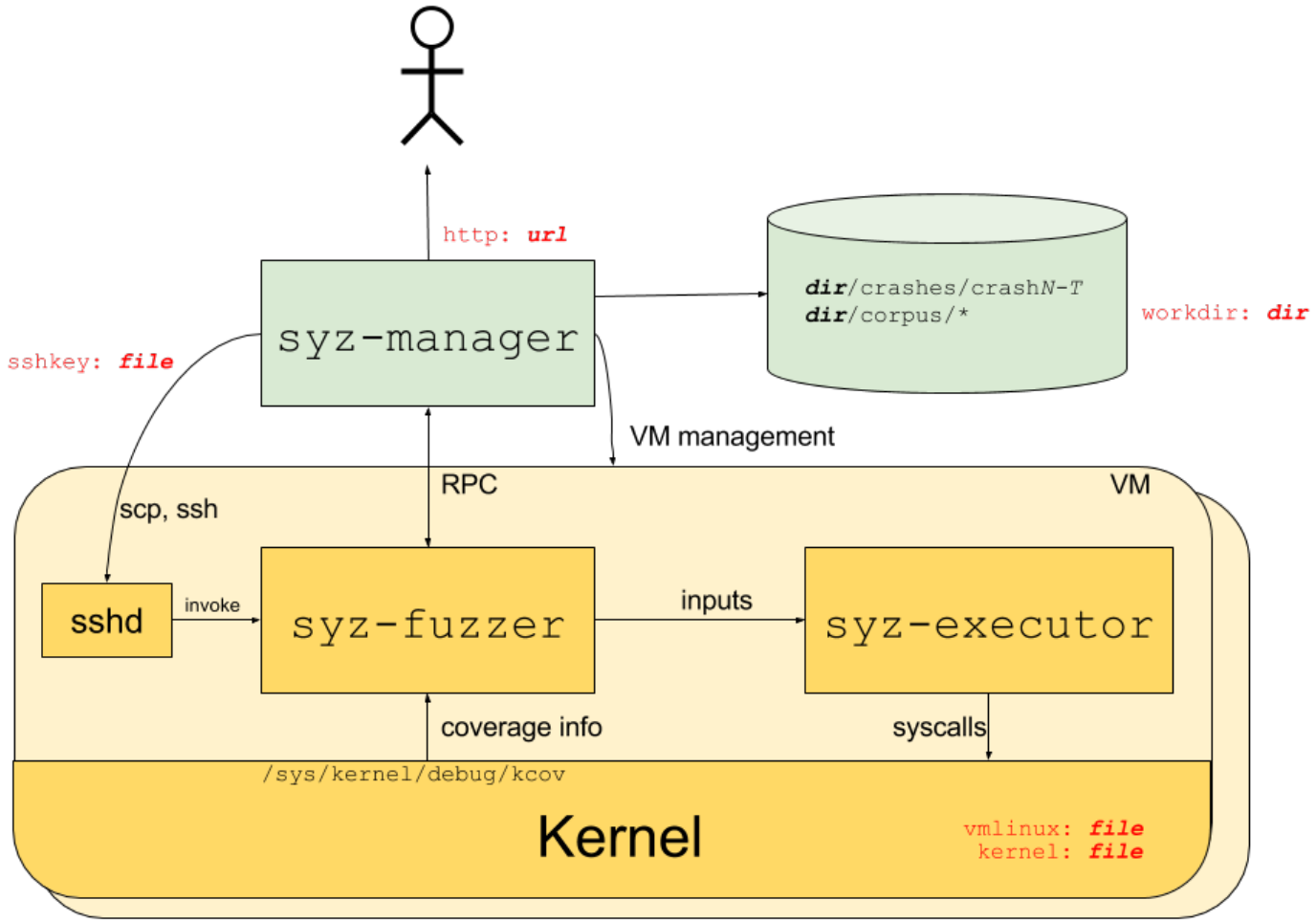
- Before fuzzing
  - Target – crash the kernel → **VM**
  - Start-up the VMs, control the VMs, start fuzzing → **Automatically?**
- During fuzzing
  - Coverage information → Fuzz **deeper?**
  - Feedback for input generation → **Unsupervised Mutation?**
- After crash
  - Reproduce → **Automatically?**
  - Minimize → **Automatically?**

Syzkaller

- Before fuzzing
  - Unsupervised
  - Multi-
    - OS (Linux, *BSD, Windows, …)
    - Arch (x86-64, arm64, …)
    - Machine (QEMU, GCE, Android phones, …)
- During fuzzing
  - Coverage-guided
  - Input Generation
- After crash
  - Generate C/syzkaller program

Architecture

https://github.com/google/syzkaller/blob/master/docs/internals.md



- syz-manager
  - Start/Monitor/Restart VMs
  - Record crashes
  - User Interface
- syz-fuzzer
  - Input generation, mutation, minimization
  - Start syz-executor process
- syz-executor
  - Execute a single input (a sequence of syscalls)

Set up

How to set up syzkaller

Below are the generic instructions for how to set up syzkaller to fuzz the Linux kernel. Instructions for a particular VM type or kernel arch can be found on these pages:

- Setup: Ubuntu host, QEMU vm, x86-64 kernel
- Setup: Ubuntu host, Odroid C2 board, arm64 kernel
- Setup: Linux host, QEMU vm, arm64 kernel
- Setup: Linux host, QEMU vm, arm kernel
- Setup: Linux host, Android device, arm64 kernel
- Setup: Ubuntu host, Android device, arm32 kernel
- Setup: Linux isolated host

Coverage

- CONFIG_KCOV
- GCC/Clang inserts a function call into every basic block
- Kernel debugfs extension collects and exposes coverage per-thread

```
if (...) {
    ...
}
```

```
__sanitizer_cov_trace_pc();     // 1
if (...) {
    __sanitizer_cov_trace_pc(); // 2
    ...
}
__sanitizer_cov_trace_pc();     // 3
```

Algorithm

1. Start with an empty corpus of programs
2. Generate a new program / choose one from corpus and mutate it
3. Run the program, collect coverage
4. Cover new code → Minimize the program and add to the corpus
5. Goto 1

Input Generation

https://github.com/google/syzkaller/blob/master/docs/syscall_descriptions_syntax.md

- Syscall Descriptions Syntax

```
syscallname "(" [arg ["," arg]*] ")" [type]
arg = argname type
argname = identifier
type = typename [ "[" type-options "]" ]
typename = "const" | "intN" | "intptr" | "flags" | "array" | "ptr" |
           "string" | "strconst" | "filename" | "len" |
           "bytesize" | "bytesizeN" | "bitsize" | "vma" | "proc"
type-options = [type-opt ["," type-opt]]
```

- Resources

```
resource fd[int32]: 0xffffffffffffffff, AT_FDCWD, 1000000
resource sock[fd]
resource sock_unix[sock]

socket(...) sock
accept(fd sock, ...) sock
listen(fd sock, backlog int32)
```

Input Generation

- Syscall Descriptions → symbolic constants

  Generates a small C program that:

  - Includes kernel headers referenced by include directives
  - Defines macros as specified by define directives
  - Prints values of symbolic constants

Input Generation

fs_ext4.txt | fs_ext4_amd64.const | fs_ext4_386.const | amd64.go | sys.txt

sys ▸ linux ▸ sys.txt

```
type signalno int32[0:65]

syz_execute_func(text ptr[in, text[target]])

open(file ptr[in, filename], flags flags[open_flags], mode flags[open_mode]) fd
# Just so that we have something that creates fd_dir resources.
open$dir(file ptr[in, filename], flags flags[open_flags], mode flags[open_mode]) fd_dir
openat$dir(fd const[AT_FDCWD], file ptr[in, filename], flags flags[open_flags], mode flags[open_mode]) fd_
openat(fd fd_dir[opt], file ptr[in, filename], flags flags[open_flags], mode flags[open_mode]) fd
creat(file ptr[in, filename], mode flags[open_mode]) fd
close(fd fd)
read(fd fd, buf buffer[out], count len[buf])
pread64(fd fd, buf buffer[out], count len[buf], pos fileoff)
readv(fd fd, vec ptr[in, array[iovec_out]], vlen len[vec])
preadv(fd fd, vec ptr[in, array[iovec_out]], vlen len[vec], off fileoff)
write(fd fd, buf buffer[in], count len[buf])
pwrite64(fd fd, buf buffer[in], count len[buf], pos fileoff)
writev(fd fd, vec ptr[in, array[iovec_in]], vlen len[vec])
pwritev(fd fd, vec ptr[in, array[iovec_in]], vlen len[vec], off fileoff)
lseek(fd fd, offset fileoff, whence flags[seek_whence])

dup(oldfd fd) fd
dup2(oldfd fd, newfd fd) fd
dup3(oldfd fd, newfd fd, flags flags[dup_flags]) fd

pipe(pipefd ptr[out, pipefd])
pipe2(pipefd ptr[out, pipefd], flags flags[pipe_flags])

tee(fdin fd, fdout fd, len intptr, f flags[splice_flags])
splice(fdin fd, offin ptr[in, fileoff[int64]], fdout fd, offout ptr[in, fileoff[int64]], len intptr, f fla
vmsplice(fd fd, vec ptr[in, array[iovec_in]], vlen len[vec], f flags[splice_flags])
```

master | 21 27 | Dmitry Vyukov, 3 years ago | 行 340，列 17 (已选择7) | 制表符长度: 4 | UTF-8 | CRLF | 纯文本 | 1

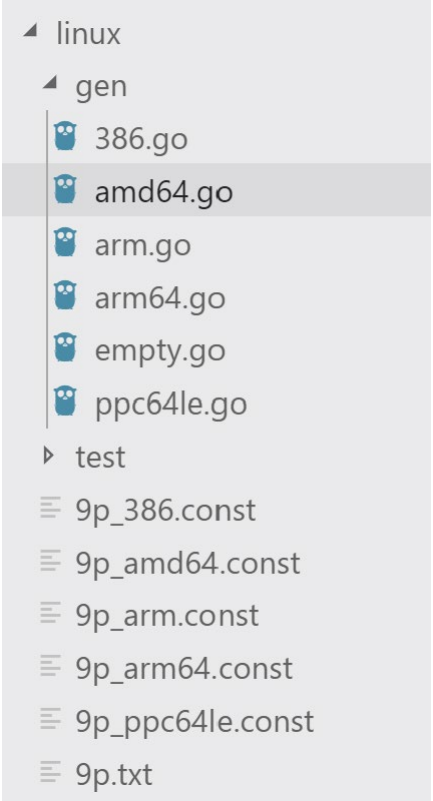
Input Generation

- Syscall Descriptions → symbolic constants

  Generates a small C program that:

  - Includes kernel headers referenced by include directives
  - Defines macros as specified by define directives
  - Prints values of symbolic constants

- Translation of descriptions into Go code



Input Generation

资源管理器 | fs_ext4.txt | fs_ext4_amd64.const | fs_ext4_386.const | amd64.go

sys ▸ linux ▸ gen ▸ amd64.go ▸ {} gen

打开的编辑器
- fs_ext4.txt sys\linux
- fs_ext4_amd64.const sys...
- fs_ext4_386.const sys\linux
- amd64.go sys\linux\gen

SYZKALLER
- fuchsia
- linux
  - gen
    - 386.go
    - amd64.go
    - arm.go
    - arm64.go
    - empty.go
    - ppc64le.go
  - test
  - 9p_386.const
  - 9p_amd64.const
  - 9p_arm.const
  - 9p_arm64.const
  - 9p_ppc64le.const
  - 9p.txt
  - aio_386.const
  - aio_amd64.const

大纲
- {} gen package
  - init function
  - consts_amd64 variable
  - resources_amd64 variable
  - revision_amd64 variable

MAVEN 项目

Denis Efremov, 10 days ago | 22 authors (Dmitry Vyukov and others)

```
1   // AUTOGENERATED FILE
2   // +build !codeanalysis
3   // +build !syz_target syz_target,syz_os_linux,syz_arch_amd64
4
```

Denis Efremov, 10 days ago | 22 authors (Dmitry Vyukov and others)

```
5   package gen
6
7   import . "github.com/google/syzkaller/prog"
8   import . "github.com/google/syzkaller/sys/linux"
9
10  func init() {
11      RegisterTarget(&Target{OS: "linux", Arch: "amd64", Revision: revision_amd64, PtrS
12  }
13
14  var resources_amd64 = []*ResourceDesc{…
233 }
234
235 var structDescs_amd64 = []*KeyedStruct{…
29640 }
29641
29642 var syscalls_amd64 = []*Syscall{…
44454 }
44455
44456 var consts_amd64 = []ConstValue{…
52264 }
52265
52266 const revision_amd64 = "036bfe5c68bb7bfe0cc9b16821d139bf4e784047"      Denis Efremov,
52267
```

master | 21 27 | Denis Efremov, 10 days ago | 行 52266，列 66 | 制表符长度: 4 | UTF-8 | CRLF | Go | 1

Input Generation

fs_ext4.txt | fs_ext4_amd64.const | fs_ext4_386.const | amd64.go

sys ▸ linux ▸ gen ▸ amd64.go ▸ {} gen

```
235  var structDescs_amd64 = []*KeyedStruct{…
29640 }
29641
29642 var syscalls_amd64 = []*Syscall{
29643     {NR: 43, Name: "accept", CallName: "accept", Args: []Type{
29644         &ResourceType{TypeCommon: TypeCommon{TypeName: "sock", FldName: "fd", TypeSize: 4}},
29645         &PtrType{TypeCommon: TypeCommon{TypeName: "ptr", FldName: "peer", TypeSize: 8, IsOptional: true}
29646         &PtrType{TypeCommon: TypeCommon{TypeName: "ptr", FldName: "peerlen", TypeSize: 8}, Type: &LenTyp
29647     }, Ret: &ResourceType{TypeCommon: TypeCommon{TypeName: "sock", FldName: "ret", TypeSize: 4, ArgDir:
29648     {NR: 43, Name: "accept$alg", CallName: "accept", Args: []Type{
29649         &ResourceType{TypeCommon: TypeCommon{TypeName: "sock_alg", FldName: "fd", TypeSize: 4}},
29650         &ConstType{IntTypeCommon: IntTypeCommon{TypeCommon: TypeCommon{TypeName: "const", FldName: "peer
29651         &ConstType{IntTypeCommon: IntTypeCommon{TypeCommon: TypeCommon{TypeName: "const", FldName: "peer
29652     }, Ret: &ResourceType{TypeCommon: TypeCommon{TypeName: "sock_algconn", FldName: "ret", TypeSize: 4,
29653     {NR: 43, Name: "accept$ax25", CallName: "accept", Args: []Type{
29654         &ResourceType{TypeCommon: TypeCommon{TypeName: "sock_ax25", FldName: "fd", TypeSize: 4}},
29655         &PtrType{TypeCommon: TypeCommon{TypeName: "ptr", FldName: "peer", TypeSize: 8, IsOptional: true}
29656         &PtrType{TypeCommon: TypeCommon{TypeName: "ptr", FldName: "peerlen", TypeSize: 8}, Type: &LenTyp
29657     }, Ret: &ResourceType{TypeCommon: TypeCommon{TypeName: "sock_ax25", FldName: "ret", TypeSize: 4, Arg
29658     {NR: 43, Name: "accept$inet", CallName: "accept", Args: []Type{
29659         &ResourceType{TypeCommon: TypeCommon{TypeName: "sock_in", FldName: "fd", TypeSize: 4}},
29660         &PtrType{TypeCommon: TypeCommon{TypeName: "ptr", FldName: "peer", TypeSize: 8, IsOptional: true}
29661         &PtrType{TypeCommon: TypeCommon{TypeName: "ptr", FldName: "peerlen", TypeSize: 8}, Type: &LenTyp
29662     }, Ret: &ResourceType{TypeCommon: TypeCommon{TypeName: "sock_in", FldName: "ret", TypeSize: 4, ArgDi
29663     {NR: 43, Name: "accept$inet6", CallName: "accept", Args: []Type{
29664         &ResourceType{TypeCommon: TypeCommon{TypeName: "sock_in6", FldName: "fd", TypeSize: 4}},
29665         &PtrType{TypeCommon: TypeCommon{TypeName: "ptr", FldName: "peer", TypeSize: 8, IsOptional: true}
29666         &PtrType{TypeCommon: TypeCommon{TypeName: "ptr", FldName: "peerlen", TypeSize: 8}, Type: &LenTyp
29667     }, Ret: &ResourceType{TypeCommon: TypeCommon{TypeName: "sock_in6", FldName: "ret", TypeSize: 4, ArgD
29668     {NR: 43, Name: "accept$ipx", CallName: "accept", Args: []Type{
29669         &ResourceType{TypeCommon: TypeCommon{TypeName: "sock_ipx", FldName: "fd", TypeSize: 4}},
29670         &PtrType{TypeCommon: TypeCommon{TypeName: "ptr", FldName: "peer", TypeSize: 8, IsOptional: true}
```

master | 21 27 | Denis Efremov, 10 days ago | 行 52266，列 66 | 制表符长度: 4 | UTF-8 | CRLF | Go | 1

Input Generation

- Syscall Descriptions → symbolic constants

  Generates a small C program that:

  - Includes kernel headers referenced by include directives
  - Defines macros as specified by define directives
  - Prints values of symbolic constants

- Translation of descriptions into Go code

- Program

```
mmap(&(0x7f0000000000), (0x1000), 0x3, 0x32, -1, 0)
r0 = open(&(0x7f0000000000)="./file0", 0x3, 0x9)
read(r0, &(0x7f0000000000), 42)
close(r0)
```

Recovery

https://syzkaller.appspot.com

https://syzkaller.appspot.com/bug?id=4462b73f1e909dcd0f6e0d5d2f0b3e8be5191b69

syzbot

Linux ▾

sign-in

memory leak in inet6_create
Status: fixed on 2019/07/10 21:40
Reported-by: syzbot+@syzkaller.appspotmail.com
Fix commit: 522924b5 net: correct udp zerocopy refcnt also when zerocopy only on append
First crash: 38d, last: 35d

Sample crash report:

```
executing program
BUG: memory leak
unreferenced object 0xffff888124365a40 (size 1280):
  comm "syz-executor032", pid 7039, jiffies 4294952820 (age 37.550s)
  hex dump (first 32 bytes):
    00 00 00 00 00 00 00 00 80 dd 7b f4 00 00 00 00  ..........{.....
    0a 00 07 40 00 00 00 00 00 00 00 00 00 00 00 00  ...@............
  backtrace:
    [<00000000dba9ec2e>] kmemleak_alloc_recursive include/linux/kmemleak.h:55 [inline]
    [<00000000dba9ec2e>] slab_post_alloc_hook mm/slab.h:439 [inline]
    [<00000000dba9ec2e>] slab_alloc mm/slab.c:3326 [inline]
    [<00000000dba9ec2e>] kmem_cache_alloc+0x134/0x270 mm/slab.c:3488
    [<0000000018c52d6d>] sk_prot_alloc+0x41/0x170 net/core/sock.c:1596
    [<00000000d9c0caab>] sk_alloc+0x35/0x2f0 net/core/sock.c:1656
    [<00000000700b302a>] inet6_create net/ipv6/af_inet6.c:180 [inline]
    [<00000000700b302a>] inet6_create+0x115/0x4b0 net/ipv6/af_inet6.c:107
    [<00000000ce38b1e5>] __sock_create+0x164/0x250 net/socket.c:1424
    [<000000007e6d40fd>] sock_create net/socket.c:1475 [inline]
    [<000000007e6d40fd>] __sys_socket+0x69/0x110 net/socket.c:1517
    [<0000000081e1179a>] __do_sys_socket net/socket.c:1526 [inline]
    [<0000000081e1179a>] __se_sys_socket net/socket.c:1524 [inline]
    [<0000000081e1179a>] __x64_sys_socket+0x1e/0x30 net/socket.c:1524
    [<000000008d9383e5>] do_syscall_64+0x76/0x1a0 arch/x86/entry/common.c:301
    [<00000000af94d8fe>] entry_SYSCALL_64_after_hwframe+0x44/0xa9

BUG: memory leak
```

All crashes (2):

Manager	Time	Kernel	Commit	Syzkaller	Config	Log	Report	Syz repro	C repro	Maintainers
ci-upstream-gce-leak	2019/06/06 17:54	upstream	156c0591	698773cb	.config	log	report	syz	C	ast@kernel.org, bpf@vg...
ci-upstream-gce-leak	2019/06/09 12:52	upstream	d1fdb6d8	0159583c	.config	log	report	syz	C	ast@kernel.org, bpf@vg...

Running Example (Ubuntu host, QEMU vm, x86-64 kernel)

- GCC
- Kernel
  - Linux 5.1.16
  - Enable options in default configs
  - Compile the kernel

```
  LD [M]  net/netfilter/xt_addrtype.ko
  LD [M]  net/netfilter/xt_mark.ko
  LD [M]  net/netfilter/xt_nat.ko
  MKPIGGY arch/x86/boot/compressed/piggy.S
  AS      arch/x86/boot/compressed/piggy.o
  LD      arch/x86/boot/compressed/vmlinux
  ZOFFSET arch/x86/boot/zoffset.h
  OBJCOPY arch/x86/boot/vmlinux.bin
  AS      arch/x86/boot/header.o
  LD      arch/x86/boot/setup.elf
  OBJCOPY arch/x86/boot/setup.bin
  BUILD   arch/x86/boot/bzImage
Setup is 16028 bytes (padded to 16384 bytes).
System is 19829 kB
CRC f3056af5
Kernel: arch/x86/boot/bzImage is ready  (#2)
```

```
# CONFIG_TEST_KASAN is not set
CONFIG_ARCH_HAS_KCOV=y
CONFIG_CC_HAS_SANCOV_TRACE_PC=y
CONFIG_KCOV=y
CONFIG_KCOV_INSTRUMENT_ALL=y
# CONFIG_DEBUG_SHIRQ is not set

#
# Debug Lockups and Hangs
#
# CONFIG_SOFTLOCKUP_DETECTOR is not set
```

Running Example (Ubuntu host, QEMU vm, x86-64 kernel)

- Image
  - debootstrap
  - *debootstrap is a tool which will install a Debian base system into a subdirectory of another, already installed system.*

```
# Create a minimal Debian distribution in a directory.
DIR=chroot
PREINSTALL_PKGS=openssh-server,curl,tar,gcc,libc6-dev,time,strace,sudo,
y,selinux-policy-default
```

```
sudo rm -rf $DIR
mkdir -p $DIR
sudo debootstrap --include=$PREINSTALL_PKGS $RELEASE $DIR http://mirrors.ustc.edu.cn/debian/
```

Running Example (Ubuntu host, QEMU vm, x86-64 kernel)

- Image

```
# Set some defaults and enable promtless ssh to the machine for root.
sudo sed -i '/^root/ { s/:x:/::/ }' $DIR/etc/passwd
echo 'T0:23:respawn:/sbin/getty -L ttyS0 115200 vt100' | sudo tee -a $DIR/etc/inittab
printf '\nauto eth0\niface eth0 inet dhcp\n' | sudo tee -a $DIR/etc/network/interfaces
echo '/dev/root / ext4 defaults 0 0' | sudo tee -a $DIR/etc/fstab
echo 'debugfs /sys/kernel/debug debugfs defaults 0 0' | sudo tee -a $DIR/etc/fstab
echo 'securityfs /sys/kernel/security securityfs defaults 0 0' | sudo tee -a $DIR/etc/fstab
echo 'configfs /sys/kernel/config/ configfs defaults 0 0' | sudo tee -a $DIR/etc/fstab
echo 'binfmt_misc /proc/sys/fs/binfmt_misc binfmt_misc defaults 0 0' | sudo tee -a $DIR/etc/fstab
echo "kernel.printk = 7 4 1 3" | sudo tee -a $DIR/etc/sysctl.conf
echo 'debug.exception-trace = 0' | sudo tee -a $DIR/etc/sysctl.conf
echo "net.core.bpf_jit_enable = 1" | sudo tee -a $DIR/etc/sysctl.conf
echo "net.core.bpf_jit_kallsyms = 1" | sudo tee -a $DIR/etc/sysctl.conf
echo "net.core.bpf_jit_harden = 0" | sudo tee -a $DIR/etc/sysctl.conf
echo "kernel.softlockup_all_cpu_backtrace = 1" | sudo tee -a $DIR/etc/sysctl.conf
echo "kernel.kptr_restrict = 0" | sudo tee -a $DIR/etc/sysctl.conf
echo "kernel.watchdog_thresh = 60" | sudo tee -a $DIR/etc/sysctl.conf
echo "net.ipv4.ping_group_range = 0 65535" | sudo tee -a $DIR/etc/sysctl.conf
echo -en "127.0.0.1\tlocalhost\n" | sudo tee $DIR/etc/hosts
echo "nameserver 8.8.8.8" | sudo tee -a $DIR/etc/resolve.conf
echo "syzkaller" | sudo tee $DIR/etc/hostname
ssh-keygen -f $RELEASE.id_rsa -t rsa -N ''
sudo mkdir -p $DIR/root/.ssh/
cat $RELEASE.id_rsa.pub | sudo tee $DIR/root/.ssh/authorized_keys
```

Running Example (Ubuntu host, QEMU vm, x86-64 kernel)

- Prepare QEMU

```
qemu-system-x86_64 \
  -kernel $KERNEL/arch/x86/boot/bzImage \
  -append "console=ttyS0 root=/dev/sda debug earlyprintk=serial slub_debug=QUZ"\
  -hda $IMAGE/stretch.img \
  -net user,hostfwd=tcp::10021-:22 -net nic \
  -enable-kvm \
  -nographic \
  -m 2G \
  -smp 2 \
  -pidfile vm.pid \
  2>&1 | tee vm.log
```

Running Example (Ubuntu host, QEMU vm, x86-64 kernel)

```
[  OK  ] Started Getty on tty4.
[  OK  ] Started Getty on tty3.
[  OK  ] Started Serial Getty on ttyS0.
[  OK  ] Started Getty on tty1.
[  OK  ] Started Getty on tty5.
[  OK  ] Started Getty on tty6.
[  OK  ] Started Getty on tty2.
[  OK  ] Reached target Login Prompts.
[  OK  ] Started OpenBSD Secure Shell server.
[  OK  ] Reached target Multi-User System.
[  OK  ] Reached target Graphical Interface.
         Starting Update UTMP about System Runlevel Changes...
[  OK  ] Started Update UTMP about System Runlevel Changes.
[  OK  ] Started Daily apt download activities.
         Starting Daily apt upgrade and clean activities...
[  OK  ] Started Daily apt upgrade and clean activities.

Debian GNU/Linux 9 syzkaller ttyS0

syzkaller login: root
Unable to get valid context for root
Last login: Sun Jul  7 01:26:51 UTC 2019 on ttyS0
Linux syzkaller 5.1.16 #2 SMP Mon Jul 15 17:03:48 CST 2019 x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
root@syzkaller:~#
root@syzkaller:~#
```

Running Example (Ubuntu host, QEMU vm, x86-64 kernel)

- Prepare QEMU
  - Start the VM
  - Test SSH connection from host to VM

```
ssh -i $IMAGE/stretch.id_rsa -p 10021 -o "StrictHostKeyChecking no" root@localhost
```

Running Example (Ubuntu host, QEMU vm, x86-64 kernel)

- syzkaller configuration

```
{
    "target": "linux/amd64",
    "http": "127.0.0.1:56741",
    "workdir": "$GOPATH/src/github.com/google/syzkaller/workdir",
    "kernel_obj": "$KERNEL",
    "image": "$IMAGE/stretch.img",
    "sshkey": "$IMAGE/stretch.id_rsa",
    "syzkaller": "$GOPATH/src/github.com/google/syzkaller",
    "procs": 8,
    "type": "qemu",
    "vm": {
        "count": 4,
        "kernel": "$KERNEL/arch/x86/boot/bzImage",
        "cpu": 2,
        "mem": 2048
    }
}
```

Running Example (Ubuntu host, QEMU vm, x86-64 kernel)

- syzkaller start up

```
mkdir workdir
sudo ./bin/syz-manager -config=my.cfg
```

```
2019/07/15 19:10:21 loading corpus...
2019/07/15 19:10:21 serving http on http://127.0.0.1:56741
2019/07/15 19:10:21 serving rpc on tcp://[::]:43055
2019/07/15 19:10:21 booting test machines...
2019/07/15 19:10:21 wait for the connection from test machine...
2019/07/15 19:10:35 machine check:
2019/07/15 19:10:35 syscalls                : 1390/2733
2019/07/15 19:10:35 code coverage           : enabled
2019/07/15 19:10:35 comparison tracing      : CONFIG_KCOV_ENABLE_COMPARISONS is not enabled
2019/07/15 19:10:35 extra coverage          : extra coverage is not supported by the kernel
2019/07/15 19:10:35 setuid sandbox          : enabled
2019/07/15 19:10:35 namespace sandbox       : /proc/self/ns/user does not exist
2019/07/15 19:10:35 Android sandbox         : enabled
2019/07/15 19:10:35 fault injection         : CONFIG_FAULT_INJECTION is not enabled
2019/07/15 19:10:35 leak checking           : CONFIG_DEBUG_KMEMLEAK is not enabled
2019/07/15 19:10:35 net packet injection    : /dev/net/tun does not exist
2019/07/15 19:10:35 net device setup        : enabled
2019/07/15 19:10:35 corpus                  : 0 (0 deleted)
2019/07/15 19:10:41 VMs 4, executed 3, cover 4217, crashes 0, repro 0
2019/07/15 19:10:51 VMs 4, executed 6330, cover 6648, crashes 0, repro 0
2019/07/15 19:11:01 VMs 4, executed 10707, cover 8320, crashes 0, repro 0
2019/07/15 19:11:11 VMs 4, executed 15851, cover 12460, crashes 0, repro 0
```

Running Example (Ubuntu host, QEMU vm, x86-64 kernel)

http://127.0.0.1:56741

syzkaller

Stats:

revision	a0626693
config	
uptime	31s
fuzzing	1m0s
corpus	266
triage queue	0
cover	11898
signal	14709
syscalls	1390
crash types	0 (0/hour)
crashes	0 (0/hour)
exec candidate	724 (40/sec)
exec fuzz	0 (0/hour)
exec gen	0 (0/hour)
exec hints	0 (0/hour)
exec minimize	3016 (167/sec)
exec seeds	0 (0/hour)
exec smash	0 (0/hour)
exec total	5107 (283/sec)
exec triage	1367 (75/sec)
executor restarts	32 (106/min)
hub: recv prog	0 (0/hour)
hub: recv prog drop	0 (0/hour)
hub: recv repro	0 (0/hour)
hub: recv repro drop	0 (0/hour)
hub: send prog add	0 (0/hour)
hub: send prog del	0 (0/hour)
hub: send repro	0 (0/hour)
manager new inputs	291 (16/sec)
suppressed	0 (0/hour)
vm restarts	4 (13/min)

Crashes:

Description	Count	Last Time	Report

Log:

```
2019/07/15 19:24:31 fuzzer vm-3 connected
2019/07/15 19:24:31 fuzzer vm-2 connected
2019/07/15 19:24:31 fuzzer vm-0 connected
2019/07/15 19:24:31 fuzzer vm-1 connected
2019/07/15 19:24:32 machine check:
2019/07/15 19:24:32 syscalls                : 1390/2733
2019/07/15 19:24:32 code coverage           : enabled
2019/07/15 19:24:32 comparison tracing      : CONFIG_KCOV_ENABLE_COMPARISONS is not enabled
2019/07/15 19:24:32 extra coverage          : extra coverage is not supported by the kernel
2019/07/15 19:24:32 setuid sandbox          : enabled
2019/07/15 19:24:32 namespace sandbox       : /proc/self/ns/user does not exist
```