linux下fuzz初试

Edvison / 2018-09-08 15:43:04 / 浏览数 4222 技术文章 技术文章 顶(0) 踩(0)

前言

本文将介绍两种fuzz工具afl、syzkaller的安装及使用。

AFL

American fuzzy lop (“afl-fuzz”)是一种通过提供随机生成的输入来测试软件，搜索能导致程序崩溃的那些输入的fuzz测试工具。
它通过对源码进行重新编译时进行插桩的方式自动产生测试用例来探索二进制程序内部新的执行路径。
与其他基于插桩技术的fuzzers相比，afl-fuzz具有较低的性能消耗，有各种高效的fuzzing策略和技巧最小化技巧，不需要先行复杂的配置，能无缝处理复杂的现实中的程序

安装afl-fuzz

从官网下载源码压缩包，解压。然后编译安装：

```
make
sudo make install
```

安装完成后一般需要配置一下，将coredumps输出为文件，而不是把崩溃信息发送到特定的处理程序：

```
echo core > /proc/sys/kernel/core_pattern
```

官方文档：http://lcamtuf.coredump.cx/afl/README.txt

测试示例

对于有源码的测试程序，可以使用afl来代替gcc或clang来编译。用afl的插桩可以优化afl-fuzz的性能，加速fuzz。
一般的编译步骤如下：

```
CC=/path/to/afl/afl-gcc ./configuer
make clean all
```

C++程序则设置为:

```
CXX=/path/to/afl/afl-g++
```

下面是一个崩溃示例程序：

```
#include <stdio.h>
#include <signal.h>

int main(int argc, char *argv[])
{
char buf[233] = {0};
FILE *input = NULL;
input = fopen(argv[1], "r");
if(input != 0)
{
 fscanf(input ,"%s", &buf);
 printf("buf is %s\n", buf);
 func(buf);
 fclose(input);
}
else
 printf("error!");

return 0;
}

int func(char *data)
{
if(data[0] == 'A')
 raise(SIGSEGV);
```

```
else
 printf("ok\n");

return 0;
}
```

当输入文件的首字母为A时程序崩溃。

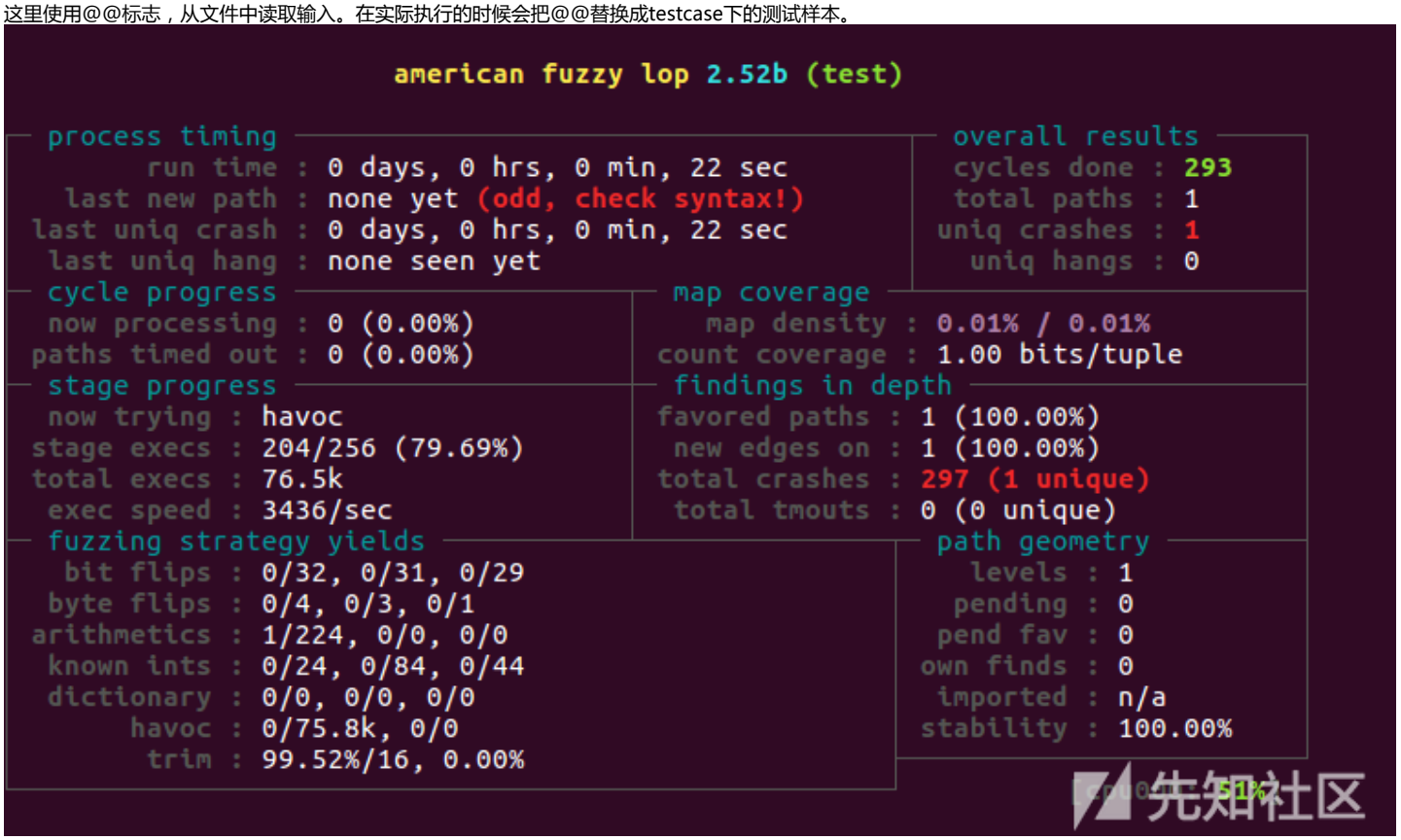
- 用afl-gcc编译程序，插桩：

  ```
  ./afl-gcc test.c -o test
  ```

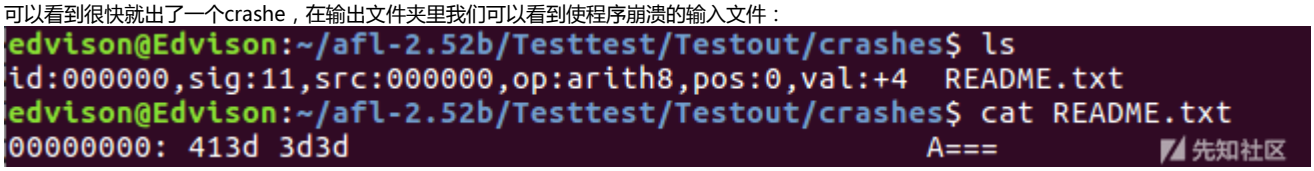
  测试
  testcase是自带的输入目录，Testout是指定的输出目录

  ```
  ./afl-fuzz -i testcase -o Testout ./test @@
  ```

- 这里使用@@标志，从文件中读取输入。在实际执行的时候会把@@替换成testcase下的测试样本。



可以看到很快就出了一个crashe，在输出文件夹里我们可以看到使程序崩溃的输入文件：



符合我们的程序崩溃逻辑。

Kernel fuzz -- syzkaller

syzkaller是一款无监督的覆盖引导内核模糊器。 Linux内核模糊测试支持最多，akaros，freebsd，fuchsia，netbsd，windows和gvisor都有不同程度的支持。我们将用它来进行linux内核模糊测试。

官方文档：https://github.com/google/syzkaller

编译内核

具体编译不再赘述(可看我这篇笔记)，注意开启下面几个选项就行：

```
CONFIG_KCOV=y
CONFIG_DEBUG_INFO=y
CONFIG_KASAN=y
CONFIG_KASAN_INLINE=y
```

安装Go语言编译器编译syzkaller，设置环境变量

```
wget https://storage.googleapis.com/golang/go1.8.1.linux-amd64.tar.gz
tar -xf go1.8.1.linux-amd64.tar.gz
mv go goroot
export GOROOT=`pwd`/goroot
export PATH=$GOROOT/bin:$PATH
mkdir gopath
export GOPATH=`pwd`/gopath
```

安装syzkaller

```
go get -u -d github.com/google/syzkaller/...
cd gopath/src/github.com/google/syzkaller/
mkdir workdir
make
```

用Debian-wheezy制作一个镜像

安装debootstrap：

```
sudo apt-get install debootstrap
```

用下面的脚本制作镜像：

```
cd gopath/src/github.com/google/syzkaller/tools/
./create-image.sh
```

然后是漫长的等待……成功后会在当前目录下看到ssh公钥和私钥文件，以及我们需要的镜像：

```
wheezy.id_rsa
wheezy.id_rsa.pub
wheezy.img
```

qemu启动

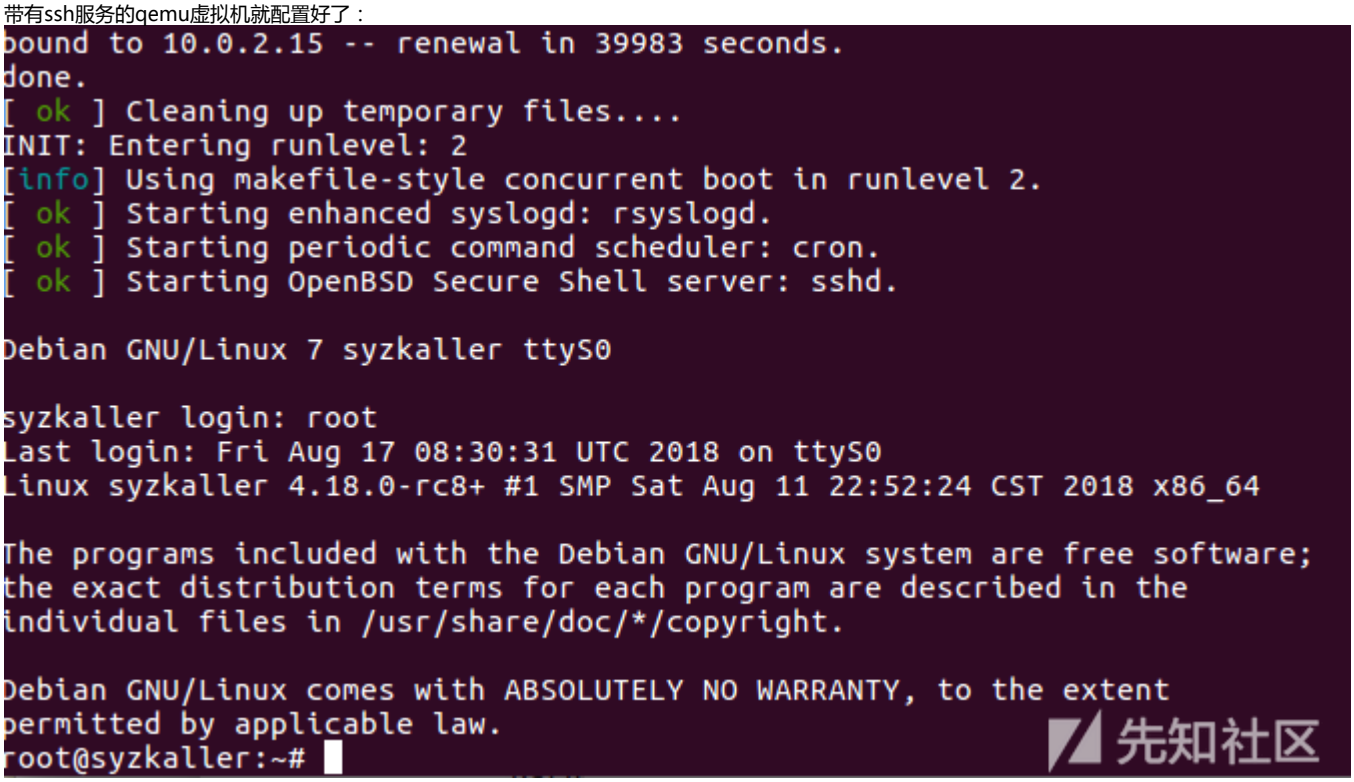
设置环境变量：

```
export KERNEL=/home/edvison/linux-kernel
export IMG=/home/edvison/gopath/src/github.com/google/syzkaller/tools
```

启动脚本：

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

带有ssh服务的qemu虚拟机就配置好了：



要结束当前的qemu实例用下面命令：

```
sudo kill $(cat vm.pid)
```

配置无密码登录ssh

官方那个ssh登录设置有点迷...用下面的方法好了。
用`ssh-keygen -t rsa`命令在本机生成私钥id_rsa和公钥id_rsa.pub。然后把公钥id_rsa.pub复制到虚拟机上。
修改虚拟机上ssh的配置：/etc/ssh/sshd_config

```
PermitRootLogin without-password

RSAAuthentication yes
PubkeyAuthentication yes
AuthorizedKeysFile  /root/.ssh/id_rsa.pub

RhostsRSAAuthentication yes
PermitEmptyPasswords no
PasswordAuthentication no

UsePAM no
```

重启ssh服务：`/etc/init.d/ssh restart`

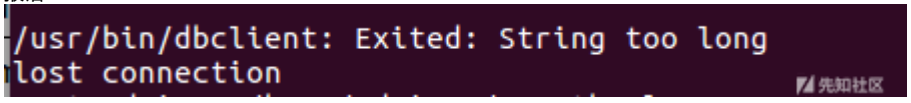
然后在本机上运行下面命令就能无密码登录ssh：

```
ssh -p 10021 -i id_rsa root@127.0.0.1
```

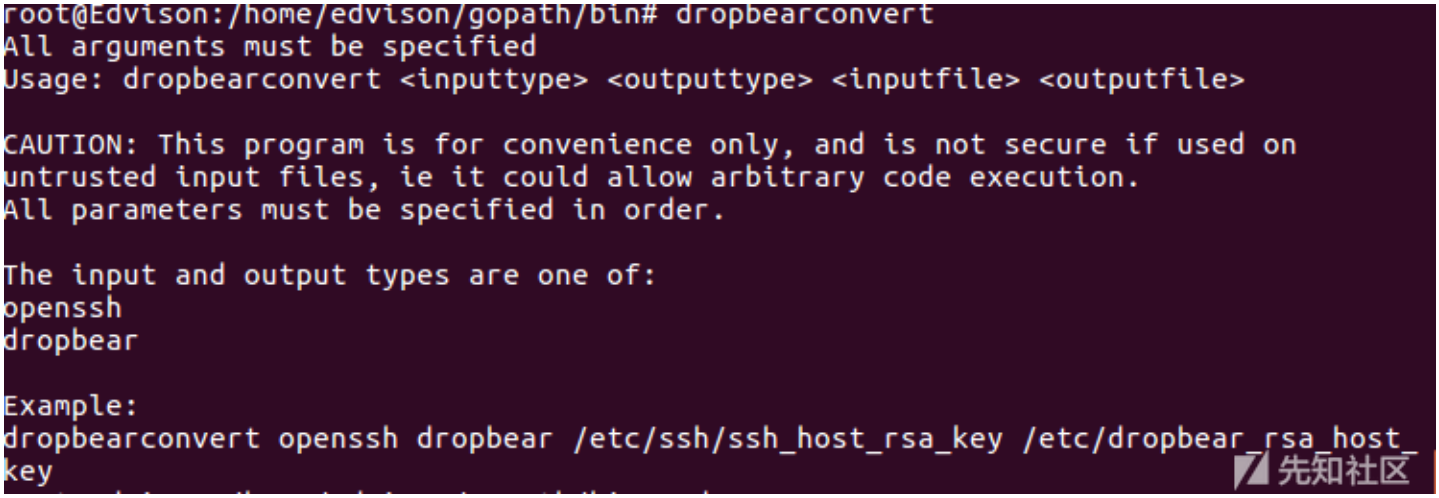
把syzkaller二进制文件复制到虚拟机中

```
scp -P 10021 -i id_rsa -r $(GOPATH)/bin root@127.0.0.1
```

报错：



搜了一下，发现是dropbear的key格式和openssh不同，要用dropbearconvert转换一下：
安装：`sudo apt install dropbear`
使用方法：




```
$ dropbearconvert openssh dropbear id_rsa id_rsa_dropbear
```

然后就可以用转换过的私钥来连接传输文件了：

```
scp -P 10021 -i id_rsa_dropbear -r $(GOPATH)/bin root@127.0.0.1
```

启动syzkaller

编辑配置文件my.cfg：

```
{
"target": "linux/amd64",
"http": "127.0.0.1:10233",
"workdir": "$GOPATH/src/github.com/google/syzkaller/workdir",
"kernel_obj": "$KERNEL",
"image": "$IMG/wheezy.img",
"sshkey": "$GOPATH/src/github.com/google/syzkaller/workdir/id_rsa_dropbear",
"syzkaller": "$GOPATH/src/github.com/google/syzkaller",
"procs": 4,
"type": "qemu",
"vm": {
 "count": 4,
 "kernel": "$KERNEL/arch/x86_64/boot/bzImage",
 "cpu": 2,
 "mem": 2048
}
}
```

运行：

```
./bin/syz-manager -config=my.cfg
```

之后打开浏览器输入127.0.0.1:10233就能看到一个包括系统调用的覆盖量，syzkaller生成的程序数量,内核被crash的日志报告等的调试信息。

上一篇：CHAINSHOT恶意软件： CV... 下一篇：初探phar://
1. 1 条回复

xuanqing20**** 2019-05-25 20:59:20

请问一下大佬，有遇到QEMU启动时报错"[FAILED] Failed to start Raise network interfaces."这个问题吗？感谢

0 回复Ta

登录 后跟帖

先知社区

现在登录

热门节点

技术文章

社区小黑板

目录

RSS 关于社区 友情链接 社区小黑板