

Практическое задание 6

Фаззинг-тестирование приложения

Задачи:

1. Подготовить VM с ОС Linux.
2. Запустить в VM Docker
3. Выполнить команду `docker pull aflplusplus/aflplusplus`
4. Выбрать фаззинг-цель для тестирования (простое приложение на 'C') и запустить фаззинг-тестирование исследуемой программы.
5. Подготовить отчет со скриншотами.

Фаззинг — это метод тестирования программного обеспечения, который используется для обнаружения уязвимостей и ошибок в приложениях. Этот подход включает в себя автоматическую генерацию и отправку случайных или некорректных данных (входных данных) в программу с целью выявления сбоев, исключений или других нежелательных реакций.

1. Устанавливаем Docker

```
shashin@shashinpc:~  
> sudo apt install -y docker.io  
[sudo] password for shashin:  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
The following additional packages will be installed:  
  cgroupfs-mount containerd criu libintl-perl libintl-xs-perl libmodule-find-perl libproc-processtable-perl libsort-naturally-perl  
  libterm-readkey-perl needrestart python3-protobuf runc tini  
Suggested packages:  
  containernetworking-plugins docker-doc aufs-tools btrfs-progs debootstrap rinse rootlesskit xfsprogs zfs-fuse | zfsutils-linux  
The following NEW packages will be installed:  
  cgroupfs-mount containerd criu docker.io libintl-perl libintl-xs-perl libmodule-find-perl libproc-processtable-perl  
  libsort-naturally-perl libterm-readkey-perl needrestart python3-protobuf runc tini  
0 upgraded, 14 newly installed, 0 to remove and 22 not upgraded.  
Need to get 66.9 MB of archives.  
After this operation, 270 MB of additional disk space will be used.  
Get:1 http://deb.debian.org/debian bookworm/main amd64 runc amd64 1.1.5+ds1-1+deb12u1 [2,710 kB]  
Get:2 http://deb.debian.org/debian bookworm/main amd64 containerd amd64 1.6.20-ds1-1+deb12u1 [25.9 MB]  
Get:3 http://deb.debian.org/debian bookworm/main amd64 tini amd64 0.19.0-1 [255 kB]  
Get:4 http://deb.debian.org/debian bookworm/main amd64 docker.io amd64 20.10.24+dfsg1-1+deb12u1+b1 [36.2 MB]  
Get:5 http://deb.debian.org/debian bookworm/main amd64 cgroupfs-mount all 1.4 [6,276 B]  
Get:6 http://deb.debian.org/debian bookworm/main amd64 python3-protobuf amd64 3.21.12-3 [245 kB]  
Get:7 http://deb.debian.org/debian bookworm/main amd64 criu amd64 3.17.1-2+deb12u1 [665 kB]  
Get:8 http://deb.debian.org/debian bookworm/main amd64 libintl-perl all 1.33-1 [720 kB]  
Get:9 http://deb.debian.org/debian bookworm/main amd64 libintl-xs-perl amd64 1.33-1 [15.6 kB]  
Get:10 http://deb.debian.org/debian bookworm/main amd64 libmodule-find-perl all 0.16-2 [10.6 kB]  
Get:11 http://deb.debian.org/debian bookworm/main amd64 libproc-processtable-perl amd64 0.634-1+b2 [43.1 kB]  
Get:12 http://deb.debian.org/debian bookworm/main amd64 libsort-naturally-perl all 1.03-4 [13.1 kB]  
Get:13 http://deb.debian.org/debian bookworm/main amd64 libterm-readkey-perl amd64 2.38-2+b1 [24.5 kB]  
Get:14 http://deb.debian.org/debian bookworm/main amd64 needrestart all 3.6-4+deb12u3 [60.5 kB]  
Fetched 66.9 MB in 4s (16.8 MB/s)  
Selecting previously unselected package runc.  
(Reading database ... 206317 files and directories currently installed.)  
Preparing to unpack .../00-runc_1.1.5+ds1-1+deb12u1_amd64.deb ...  
Unpacking runc (1.1.5+ds1-1+deb12u1) ...  
Selecting previously unselected package containerd.  
Preparing to unpack .../01-containerd_1.6.20-ds1-1+deb12u1_amd64.deb ...  
Unpacking containerd (1.6.20-ds1-1+deb12u1) ...  
Selecting previously unselected package tini.  
Preparing to unpack .../02-tini_0.19.0-1_amd64.deb ...  
Unpacking tini (0.19.0-1) ...  
Selecting previously unselected package cgroupfs-mount.  
Preparing to unpack .../03-cgroupfs-mount_1.4_all.deb ...  
Unpacking cgroupfs-mount (1.4) ...  
Selecting previously unselected package python3-protobuf.  
Preparing to unpack .../04-python3-protobuf_3.21.12-3_amd64.deb ...  
Unpacking python3-protobuf (3.21.12-3) ...  
Selecting previously unselected package criu.  
Preparing to unpack .../05-criu_3.17.1-2+deb12u1_amd64.deb ...  
Unpacking criu (3.17.1-2+deb12u1) ...  
Selecting previously unselected package libintl-perl.  
Preparing to unpack .../06-libintl-perl_1.33-1_all.deb ...  
Unpacking libintl-perl (1.33-1) ...  
Selecting previously unselected package libintl-xs-perl.  
Preparing to unpack .../07-libintl-xs-perl_1.33-1_amd64.deb ...  
Unpacking libintl-xs-perl (1.33-1) ...  
Selecting previously unselected package libmodule-find-perl.  
Preparing to unpack .../08-libmodule-find-perl_0.16-2_all.deb ...  
Unpacking libmodule-find-perl (0.16-2) ...  
Selecting previously unselected package libproc-processtable-perl.  
Preparing to unpack .../09-libproc-processtable-perl_0.634-1+b2_amd64.deb ...  
Unpacking libproc-processtable-perl (0.634-1+b2) ...  
Selecting previously unselected package libsort-naturally-perl.  
Preparing to unpack .../10-libsort-naturally-perl_1.03-4_all.deb ...  
Unpacking libsort-naturally-perl (1.03-4) ...  
Selecting previously unselected package libterm-readkey-perl.  
Preparing to unpack .../11-libterm-readkey-perl_2.38-2+b1_amd64.deb ...  
Unpacking libterm-readkey-perl (2.38-2+b1) ...  
Selecting previously unselected package needrestart.  
Preparing to unpack .../12-needrestart_3.6-4+deb12u3_all.deb ...  
Unpacking needrestart (3.6-4+deb12u3) ...  
Selecting previously unselected package docker.io.  
Preparing to unpack .../13-docker.io_20.10.24+dfsg1-1+deb12u1+b1_amd64.deb ...  
Unpacking docker.io (20.10.24+dfsg1-1+deb12u1+b1) ...  
Setting up runc (1.1.5+ds1-1+deb12u1) ...  
Setting up containerd (1.6.20-ds1-1+deb12u1) ...  
Setting up tini (0.19.0-1) ...  
Setting up cgroupfs-mount (1.4) ...  
Setting up python3-protobuf (3.21.12-3) ...  
Setting up criu (3.17.1-2+deb12u1) ...  
Setting up libintl-perl (1.33-1) ...  
Setting up libintl-xs-perl (1.33-1) ...  
Setting up libmodule-find-perl (0.16-2) ...  
Setting up libproc-processtable-perl (0.634-1+b2) ...  
Setting up libsort-naturally-perl (1.03-4) ...  
Setting up libterm-readkey-perl (2.38-2+b1) ...  
Setting up needrestart (3.6-4+deb12u3) ...  
Setting up docker.io (20.10.24+dfsg1-1+deb12u1+b1) ...  
Created symlink /etc/systemd/system/docker.service from /usr/lib/systemd/system/docker.service to /usr/lib/systemd/system/dockerd.service.  
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service from /usr/lib/systemd/system/docker.service to /usr/lib/systemd/system/dockerd.service.  
Processing triggers for libc-bin (2.36-9+deb12u1) ...
```

2. Запускаем Docker

```
> sudo systemctl enable --now docker  
Synchronizing state of docker.service with SysV service script with /lib/systemd/systemd-sysv-install.  
Executing: /lib/systemd/systemd-sysv-install enable docker  
[~]  
|
```

3. Добавляем пользователя в группу docker

```
> sudo usermod -aG docker $USER
> newgrp docker
> docker --version
Docker version 20.10.24+dfsg1, build 297e128
>
```

4. Загрузка docker образа с инструментарием для фаззинга

```
> docker pull aflplusplus/aflplusplus
Using default tag: latest
latest: Pulling from aflplusplus/aflplusplus
215ed5a63843: Pull complete
c2afcb461141: Pull complete
3ca02ff3323b: Pull complete
cada57a56dd5: Pull complete
f281e3305537: Pull complete
52b1e6541358: Pull complete
9b0d25773932: Pull complete
0fcef20a7fd1: Pull complete
6561610a9df5: Pull complete
8f974a5e3406: Pull complete
ce657232950d: Pull complete
61b461f082ab: Pull complete
Digest: sha256:23edea458bcc1e5647b19bfba205f21408235972ce6b346c326dbeef179919d6
Status: Downloaded newer image for aflplusplus/aflplusplus:latest
docker.io/aflplusplus/aflplusplus:latest
>
```

5. Запуск контейнера

```
>
> sudo docker run -ti -v ~/aflt:/src aflplusplus/aflplusplus
[AFL++ dcb7f165c8a9] /AFLplusplus #
```

6. Пишем скрипт на C для запуска фаззинга

```
GNU nano 6.2 vulnerable.c *
#include <stdio.h>
#include <string.h>

void vulnerable(char *input) {
    char buffer[16];
    strcpy(buffer, input);
}

int main(int argc, char **argv) {
    if(argc > 1) {
        vulnerable(argv[1]);
    }
    return 0;
}
```

7. Запуск

```
[AFL++ dcb7f165c8a9] /src # afl-fuzz input -o output -- ./valnerable @@
```

```

american fuzzy lop ++4.33a {main} (./vulnerable) [explore]
process timing ----- overall results -----
  run time : 0 days, 0 hrs, 0 min, 20 sec      cycles done : 1030
  last new find : none yet (odd, check syntax!) corpus count : 1
last saved crash : none seen yet              saved crashes : 0
  last saved hang : none seen yet              saved hangs : 0
cycle progress ----- map coverage -----
  now processing : 0.1030 (0.0%)                map density : 12.50% / 12.50%
  runs timed out : 0 (0.00%)                   count coverage : 449.00 bits/tuple
stage progress ----- findings in depth -----
  now trying : havoc                            favored items : 1 (100.00%)
stage execs : 99/100 (99.00%)                 new edges on : 1 (100.00%)
total execs : 103k                            total crashes : 0 (0 saved)
  exec speed : 5161/sec                       total tmouts : 0 (0 saved)
fuzzing strategy yields ----- item geometry -----
  bit flips : 0/0, 0/0, 0/0                  levels : 1
  byte flips : 0/0, 0/0, 0/0                pending : 0
arithmetics : 0/0, 0/0, 0/0                pend fav : 0
  known ints : 0/0, 0/0, 0/0                own finds : 0
  dictionary : 0/0, 0/0, 0/0, 0/0           imported : 0
havoc/splice : 0/103k, 0/0                  stability : 100.00%
py/custom/rq : unused, unused, unused, unused
  trim/eff : disabled, n/a
strategy: explore ----- state: started :-) -----

```

В случае краша, система напишет об этом (“saved crashes:”, “total crashes” справа)

```
shashinpc# pwd
/home/shashin/aflt
shashinpc# cd output
shashinpc# ls
main
shashinpc# cd main
shashinpc# ls
cmdline  crashes  fuzz_bitmap  fuzzer_setup  fuzzer_stats  hangs  is_main_node  plot_data  queue  target_hash
shashinpc#
```

Описание компонентов фаззер.

1. `queue` — место где хранятся входные данные, используемые фаззером для создания новых текстовых вариантов
2. `crashes` — каталог, в котором хранятся файлы, вызывающие сбои программы во время тестирования
3. `hangs` — папка, где хранятся тестовые файлы, которые привели к зависанию программы, то есть к длительному выполнению без завершения

4. `fuzzer_stats` — файлы, содержащие статистические данные о работе фаззера, такие как количество выполненных циклов и покрытие кода
5. `plot_data` — информация, используемая для построения графиков, например, зависимость покрытия кода от времени
6. `target_hash` — данные о хэшировании целевых файлов, которые помогают отслеживать уникальность входных данных
7. `cmdline` — запись команд, используемых для запуска фаззера, включая аргументы командной строки
8. `fuzz_bitmap` — данные, которые отслеживают покрытие кода, включая битмапы, используемые для определения новых ветвей выполнения
9. `fuzzer_setup` — конфигурационные файлы или параметры, определяющие настройки работы фаззера

Фаззинг является мощным методом тестирования. Его автоматизация и способность находить ошибки делают его незаменимым инструментом в современном процессе разработки программного обеспечения.