Docs

Simple doc for PASSFuzz (temporary name).

Design Overview

- PASSFuzz exploits a kernel module named **NoDrop** to get the syscall feedback information. The main part of fuzzer is able to fetch the syscall information and guide fuzzing.
- PASSFuzz uses the syscall patter coverage feedback mechanism as the paper describes, and does not change or modify the original AFL priority algorithm.

Modules

- **NoDrop**: The kernel module that is responsible for catching syscall information with soundness guarantees and transfer to the fuzzer.
- **Forkserver**: Same as AFL forkserver but not injected into the program. It also takes the tasks of processing feedback information.
- **Fuzzer**: The main fuzzer base.
- **LSHashing**: A Python script for a better hash algorithm, but abandoned in our latest verson due to the bad performance. A C++ version may be more suitable in the future.
- **TestPaths**: A suite of Python scripts to get relative information.
- **FunctionHook**: Hook important library functions to enrich feedback sources. Not needed in the latest version.

Structure

PASSFuzz

```
.
├── Makefile
├── afl-analyze.c
├── afl-as.c
├── afl-as.h
├── afl-cmin
├── afl-fuzz.c
├── afl-gcc.c
├── afl-gotcpu.c
├── afl-plot
├── afl-showmap.c
├── afl-sys-showmap.c                    // abandoned script in our lates version
├── afl-tmin.c
├── afl-whatsup
├── alloc-inl.h
├── begin.sh                             // record the commands used in experiments
├── calculateMD5.py                       // generate MD5 for files
├── config.h
├── debug.h                               // debugging-related header
├── dictionaries
├── docs                                  // AFL docs, not our docs
├── exit.sh                               // the script to exit, not needed in our latest version
├── forkserver.h                          // Forkserver
├── funchook                              // FunctionHook
│   ├── allfile-hooking.c
│   └── hooking.so
├── hash.h
├── include                               // NoDrop-relative headers
│   ├── common.h
│   ├── events.h
│   ├── export.h
│   └── ioctl.h
├── libdislocator
├── libtokencap
├── llvm_mode
├── logs                                  // The log files generated by debugging mode
│   ├── cur_log.txt
│   ├── cur_tuple
│   ├── cur_tuple1
│   ├── cur_tuple2
│   ├── logging.txt
│   └── tupleComp.txt
├── lsh.py                                // LSHashing
├── qemu_mode                             // original AFL-QEMU part, not needed in our tool
│   ├── build_qemu_support.sh
│   └── patches
│       ├── afl-qemu-cpu-inl.h
│       ├── configure.diff
│       ├── cpu-exec.diff
│       ├── elfload.diff
│       ├── memfd.diff
│       └── syscall.diff
├── test-instr.c
├── test_paths.py                         // TestPaths suite
├── test_readtuple
├── test_readtuple.c
├── test_readtuple.py                     // TestPaths suite
├── testcases
└── types.h

55 directories, 194 files
```

NoDrop

```
.
├── CMakeLists.txt
├── README.md
├── benchmark                             // The NoDrop benchmark, not fuzzing benchmarks
│   ├── apache2
│   │   ├── apache2_install.sh
│   │   ├── apr-1.7.0.tar.bz2
│   │   ├── apr-util-1.6.1.tar.bz2
│   │   ├── http-test-files-1.tar.xz
│   │   └── httpd-2.4.48.tar.bz2
│   ├── nginx
│   │   ├── http-test-files-1.tar.xz
│   │   ├── nginx-1.21.1.tar.gz
│   │   └── nginx_install.sh
│   ├── redis
│   │   ├── redis-6.0.9.tar.gz
│   │   └── redis_install.sh
│   ├── test_7z.py
│   ├── test_nginx.py
│   ├── test_openssl.py
│   ├── test_postmark.py
│   └── test_redis.py
├── include                               // exported headers
```

```
│   ├── common.h
│   ├── events.h
│   ├── export.h                          // same with the content in events_table.c and so on
│   └── ioctl.h
├── kmodule                               // core codes
│   ├── CMakeLists.txt
│   ├── Makefile.in
│   ├── elf.c
│   ├── events.c
│   ├── fillers.c
│   ├── fillers.h
│   ├── fillers_table.c
│   ├── flags.h
│   ├── loader.c
│   ├── nod_main.c
│   ├── nodrop.h                          // headers for hyperparamters
│   ├── privil.c
│   ├── proc.c                            // API to interate with user-space processes
│   ├── procinfo.c
│   ├── procinfo.h
│   ├── syscall.h                         // syscall header
│   ├── syscall_table.c
│   ├── tables
│   │   ├── dynamic_params_table.c
│   │   ├── events_table.c                // syscall information rules
│   │   └── flags_table.c
│   └── trace.c
├── monitor
│   ├── CMakeLists.txt
│   ├── mmheap
│   │   ├── mmheap.c
│   │   └── mmheap.h
│   ├── musl.specs
│   ├── script_x86-64.ld
│   └── src
│       ├── dynlink.h
│       ├── main.c
│       ├── pkeys.h
│       └── startup.c
├── musl
└── scripts
    ├── CMakeLists.txt
    ├── StressTesting
    │   ├── 1.txt
    │   ├── CMakeLists.txt
    │   ├── attack.c
    │   ├── attack.sh
    │   ├── stress.c
    │   └── stress.sh
    ├── ctrl                               // A testing interface
    │   ├── CMakeLists.txt
    │   └── nodrop-ctl.c
    ├── getmusl.sh
    ├── mkfig
    │   ├── CMakeLists.txt
    │   ├── draw.cpp
    │   ├── matplotlibcpp.h
    │   └── pyconfig.cmake
    ├── musl-1.2.3.tar.gz
    └── tests
        ├── CMakeLists.txt
        └── multithread.c

70 directories, 514 files
```

Installation

NoDrop is known to be working under Linux 4.15 and 5.4 kernels. Other versions have not been tested. Recommend 4.15 kernel.

Usage

- Most commands are the same with AFL，but note to change the target program binary name to toTest（or adjust according to nodrop.h）
- For example，to fuzz xpdf suite，we will change pdftotext to toTest，and the command is like:

```
./afl-fuzz -i ../xpdfTest/inputs -o ../xpdfTest/output
../xpdfTest/toTest @@ ../xpdfTest/out/null
```

- More commands are in begin.sh.