Sydr & CASR

Динамический анализ для SDL

Андрей Федотов infosec.exchange/@anfedotoff

1 декабря 2022

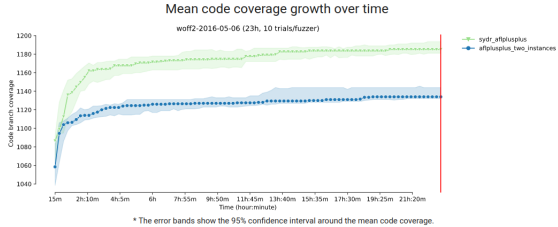
26 Июн 2020	Sydr v0.1.0. Первый релиз.
25 Сен 2020	CASR. "Иванниковские чтения".
11 Дек 2020	Sydr. "Открытая конференция ИСП РАН".
25 Сен 2021	Символьные указатели. "Иванниковские чтения".
8 Окт 2021	Sydr v1.1.0 (sydr-fuzz). Создание OSS-Sydr-Fuzz.
3 Дек 2021	Casr-cluster. Предикаты безопасности. "Открытая конференция ИСП РАН".
4 Июн 2022	CI система для фаззинга фреймворков ИИ.
18 Июл 2022	Sydr v1.8.0 поддержка AFL++ в sydr-fuzz.
23 Сен 2022	Сильно оптимистичные решения. "Иванниковские чтения".
21 Окт 2022	CASR стал открытым ПО.
2 Дек 2022	Sydr-fuzz. "Открытая конференция ИСП РАН".

Открытое ПО как база для создания продукта

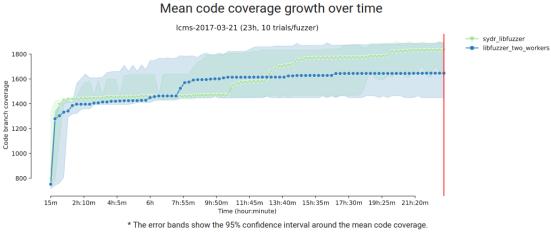
Работа с зависимостями:

- Triton — фреймворк динамического символьного выполнения. **55 PR**.
- AFL++ — **5 PR**.
- DynamoRIO — фреймворк для динамической бинарной инструментации. **6 PR**.
- LLVM(libFizzer). **1 PR**.
- goblin — Rust библиотека парсинга форматов исполняемых файлов. **6 PR**.

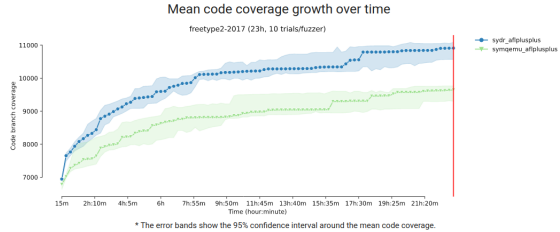
1. Sydr-fuzz достиг большего покрытия, чем другие фаззеры.
2. Sydr-fuzz победил на большем числе бенчмарков.



Sydr+AFLplusplus vs 2xAFLplusplus



Sydr+libFizzer vs 2xlibFuzzer



Sydr+AFLplusplus vs SymQEMU+AFLplusplus

sydr-fuzz.github.io/fuzzbench/

Удобство использования — ключ к успеху

ИСП РАН

Проект sydr-fuzz:

- corpus (результирующий корпус)
- crashes (найденные аварийных завершения)
- libfuzzer/aflplusplus/sydr (рабочие директории)
- casr (результаты кластеризации)
- security (предикаты безопасности)
- coverage (покрытие кода)
- sydr-fuzz*.log (логи)

Конфиг-файл:

```
[sydr]
target = "/decode_wav_sydr @@"
jobs = 2

[aflplusplus]
target = "/decode_wav_fuzz"
args = "-x wav.dict -i /corpus"
jobs = 2

[cov]
target = "/decode_wav_cov @@"
```

Запуск: *sydr-fuzz -c config.toml run|security|cmin|casr|cov-**

OSS-Sydr-Fuzz: новые ошибки в открытом ПО

ИСП РАН

github.com/ispras/oss-sydr-fuzz — это форк OSS-Fuzz для гибридного фаззинга с помощью Sydr.

- **40**+ проектов и **290**+ фаззинг-целей.
- Найдено **85** ошибок (70 подтверждены, 59 исправлены) в PyTorch, TensorFlow, OpenCV (OpenJPEG), Torchvision, Tarantool, icu, nDPI, Cairo и других проектах. Полный список трофеев тут.
- Чекеры Sydr помогают находить новые ошибки после фаззинга. Найдено **10**+ ошибок с помощью предикатов безопасности.
- Casr существенно сокращает время анализа аварийных завершений (1800+ аварийных завершений свелось к 7 ошибкам в PyTorch).

Использование Sydr & CASR

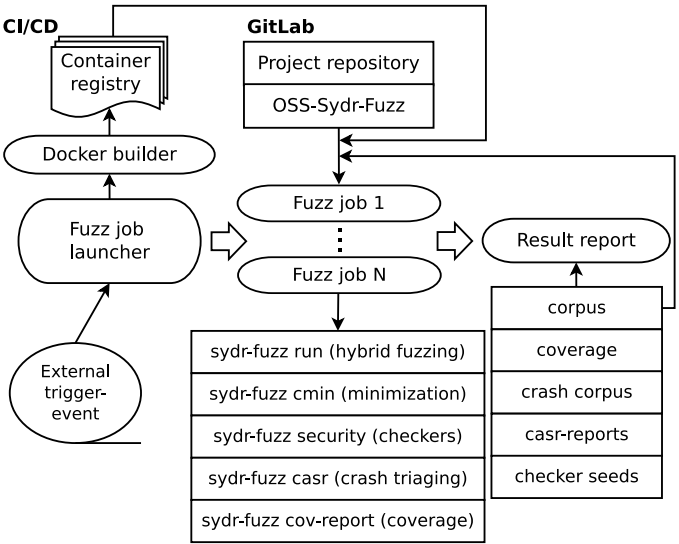
ИСП РАН

- центр Доверенного Искусственного Интелекта ИСП РАН;
- проект UEFI-прошивки Amaranth (базируется на EDK II);
- проект OSS-Sydr-Fuzz (взаимодействие с компаниями: Код Безопасности, R-Vision).

Группа пользователей (50+ участников) Sydr в Telegram: @sydr_fuzz

CI система гибридного фаззинга

ИСП РАН



Символьные предикаты безопасности

- предикаты нацелены на поиск ошибок выхода за границы массива, целочисленного переполнения, деления на ноль;
- предикаты применяются после фаззинга на минимизированном корпусе с хорошим покрытием;
- срабатывания верифицируются на санитайзерах (asan+ubsan);
- уникализация срабатываний;
- предикаты показывают 96% точность обнаружения ошибок предикатами безопасности на наборе тестов Juliet.

Чекеры находят пропущенные фаззером ошибки

ИСП РАН

github.com/opencv/opencv/issues/22284

opencv/3rdparty/openjpeg/openjp2/image.c:134:

```
l_y1 = p_cp->ty0 + (p_cp->th - 1U) * p_cp->tdy; /* can't overflow */
```

Can't overflow? But we can!

Срабатыване символьного чекера Sydr:

opj_image_comp_header_update:/opencv/3rdparty/openjpeg/openjp2/image.c:134 - imul r15d, eax - **unsigned integer overflow**

Автоматическое подтверждение дефекта санитайзерами:

/opencv/3rdparty/openjpeg/openjp2/image.c:134:40: runtime error: **unsigned integer overflow**: 2 * 4278190076 cannot be represented in type 'unsigned int'

1. Создание отчетов об аварийных завершениях: **casr-san**, **casr-gdb**;
2. Дедупликация отчетов с помощью **casr-cluster -d**;
3. Кластеризация отчетов с помощью **casr-cluster -c**;
4. Просмотр отчетов с помощью **casr-cli**.

github.com/ispras/casr

Фрагмент резюме команды sydr-fuzz casr

ИСП РАН

```
2022-11-13 14:00:39] [INFO] ==> <cl10>
2022-11-13 14:00:39] [INFO] Crash: /fuzz/sydr-fuzz-afl++-out/casr/cl10/crash-d07585811a792f15991c6ce9896f1106303ba58e
2022-11-13 14:00:39] [INFO]   casr-san: NOT_EXPLOITABLE: SourceAvNearNull: /xlnt/source/detail/serialization/xlsx_consumer.cpp:2044:22
2022-11-13 14:00:39] [INFO]   casr-gdb: NOT_EXPLOITABLE: SourceAvNearNull: /xlnt/source/detail/serialization/xlsx_consumer.cpp:2044
2022-11-13 14:00:39] [INFO]   Similar crashes: 1
2022-11-13 14:00:39] [INFO] Cluster summary -> SourceAvNearNull: 2
2022-11-13 14:00:39] [INFO] ==> <cl11>
2022-11-13 14:00:39] [INFO] Crash: /fuzz/sydr-fuzz-afl++-out/casr/cl11/crash-f403e0aedb35c05126272da86b4312939bb1efc1
2022-11-13 14:00:39] [INFO]   casr-san: NOT_EXPLOITABLE: heap-buffer-overflow(read): /xlnt/source/../source/detail/cryptography/compound_document.hpp:83:30
2022-11-13 14:00:39] [INFO]   casr-gdb: NOT_EXPLOITABLE: AbortSignal: /xlnt/third-party/utfcpp/utf8/checked.h:216
2022-11-13 14:00:39] [INFO]   Similar crashes: 1
2022-11-13 14:00:39] [INFO] Crash: /fuzz/sydr-fuzz-afl++-out/casr/cl11/crash-a02068022d89646963e409eacd5d59ea63089750
2022-11-13 14:00:39] [INFO]   casr-san: NOT_EXPLOITABLE: heap-buffer-overflow(read): /xlnt/source/../source/detail/cryptography/compound_document.hpp:83:30
2022-11-13 14:00:39] [INFO]   casr-gdb: No crash
2022-11-13 14:00:39] [INFO]   Similar crashes: 1
2022-11-13 14:00:39] [INFO] Cluster summary -> heap-buffer-overflow(read): 2 AbortSignal: 1
2022-11-13 14:00:39] [INFO] ==> <cl12>
2022-11-13 14:00:39] [INFO] Crash: /fuzz/sydr-fuzz-afl++-out/casr/cl12/crash-292e7f90b9ea11b176c04f106fe2a9e439b5b40b
2022-11-13 14:00:39] [INFO]   casr-san: PROBABLY_EXPLOITABLE: DestAvNearNull: /xlnt/source/../source/detail/binary.hpp:278:9
2022-11-13 14:00:39] [INFO]   casr-gdb: PROBABLY_EXPLOITABLE: DestAvNearNull: /xlnt/source/detail/binary.hpp:278
2022-11-13 14:00:39] [INFO]   Similar crashes: 2
2022-11-13 14:00:39] [INFO] Crash: /fuzz/sydr-fuzz-afl++-out/casr/cl12/crash-51b39d8f893faefd1d3d2003d438b82b470557e2
2022-11-13 14:00:39] [INFO]   casr-san: EXPLOITABLE: heap-buffer-overflow(write): /xlnt/source/../source/detail/binary.hpp:278:9
2022-11-13 14:00:39] [INFO]   casr-gdb: EXPLOITABLE: DestAv: /xlnt/source/detail/binary.hpp:278
2022-11-13 14:00:39] [INFO]   Similar crashes: 1
2022-11-13 14:00:39] [INFO] Cluster summary -> DestAv: 1 DestAvNearNull: 4 heap-buffer-overflow(write): 1
2022-11-13 14:00:39] [INFO] ==> <cl13>
2022-11-13 14:00:39] [INFO] Crash: /fuzz/sydr-fuzz-afl++-out/casr/cl13/crash-5ba9e014e4314b5fa1f0e270938024f7c0d02d00
2022-11-13 14:00:39] [INFO]   casr-san: EXPLOITABLE: heap-buffer-overflow(write): /xlnt/source/detail/cryptography/base64.cpp:176:32
2022-11-13 14:00:39] [INFO]   casr-gdb: No crash
2022-11-13 14:00:39] [INFO]   Similar crashes: 5
2022-11-13 14:00:39] [INFO] Crash: /fuzz/sydr-fuzz-afl++-out/casr/cl13/crash-18ebf7db76ffe9fc403faaebc0003d166e0ecc44
2022-11-13 14:00:39] [INFO]   casr-san: EXPLOITABLE: heap-buffer-overflow(write): /xlnt/source/detail/cryptography/base64.cpp:148:36
2022-11-13 14:00:39] [INFO]   casr-gdb: No crash
2022-11-13 14:00:39] [INFO]   Similar crashes: 3
2022-11-13 14:00:39] [INFO] Cluster summary -> heap-buffer-overflow(write): 8
```

Пример CASR отчета

ИСП РАН

```
Crash Report for /load_fuzzer
Severity: NOT_EXPLOITABLE: SourceAv
Crash line: /xlnt/source/detail/cryptography/compound_document.cpp:723:19

▼ Date
  ◦ 2022-09-28T14:36:46.129664+03:00
▼ Uname
  ◦ Linux titanfall 5.13.0-51-generic #58~20.04.1-Ubuntu SMP Tue Jun 14 11:29:12 UTC 2022 x86_64 x86_64 x86_64 GNU/Linux
▼ OS
  ◦ Ubuntu
▼ OSRelease
  ◦ 20.04
▼ Architecture
  ◦ amd64
▶ ExecutablePath
▼ ProcCmdline
  ◦ /load_fuzzer /fuzz/sydr-fuzz-out/crashes/crash-120697a7f5b87c03020f321c8526adf0f4bcc2dc
▼ CrashSeverity
  ◦ NOT_EXPLOITABLE
  ◦ SourceAv
  ◦ Access violation on source operand
  ◦ The target crashed on an access violation at an address matching the source operand of the current instruction. This likely indicates a read access violation.
▼ Stacktrace
  ◦ #0 0x10cfb3c in xlnt::detail::compound_document::follow_chain(int, std::vector<int, std::allocator<int> > const&) /xlnt/source/detail/cryptography/compound_document.cpp:723:19
  ◦ #1 0x10bf1fb in xlnt::detail::compound_document::read_ssat() /xlnt/source/detail/cryptography/compound_document.cpp:1282:29
  ◦ #2 0x10bd53f in xlnt::detail::compound_document::compound_document(std::istream&) /xlnt/source/detail/cryptography/compound_document.cpp:516:5
  ◦ #3 0x998b40 in (anonymous namespace)::decrypt_xlsx(std::vector<unsigned char, std::allocator<unsigned char> > const&, std::__cxx11::basic_string<char16_t, std::char_traits<char16_t>, std::allocator<char16_t>
  ◦ #4 0x99833a in xlnt::detail::decrypt_xlsx(std::vector<unsigned char, std::allocator<unsigned char> > const&, std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> > const&) /xlnt/sour
  ◦ #5 0x999aee in xlnt::detail::xlsx_consumer::read(std::istream&, std::__cxx11::basic_string<char, std::char_traits<char>, std::allocator<char> > const&) /xlnt/source/detail/cryptography/xlsx_crypto_consumer.c
  ◦ #6 0x54a288 in xlnt::workbook::load(std::istream&) /xlnt/source/workbook/workbook.cpp:901:22
  ◦ #7 0x571587 in xlnt::workbook::load(std::vector<unsigned char, std::allocator<unsigned char> > const&) /xlnt/source/workbook/workbook.cpp:919:5
  ◦ #8 0x515612 in LLVMFuzzerTestOneInput /xlnt/build/../load_fuzzer.cc:9:23
  ◦ #9 0x442481 in fuzzer::Fuzzer::ExecuteCallback(unsigned char const*, unsigned long) /llvm-project/compiler-rt/lib/fuzzer/FuzzerLoop.cpp:611:15
  ◦ #10 0x42c39c in fuzzer::RunOneTest(fuzzer::Fuzzer*, char const*, unsigned long) /llvm-project/compiler-rt/lib/fuzzer/FuzzerDriver.cpp:324:6
  ◦ #11 0x4320eb in fuzzer::FuzzerDriver(int*, char***, int (*)(unsigned char const*, unsigned long)) /llvm-project/compiler-rt/lib/fuzzer/FuzzerDriver.cpp:860:9
  ◦ #12 0x45b682 in main /llvm-project/compiler-rt/lib/fuzzer/FuzzerMain.cpp:20:10
  ◦ #13 0x7ffff7a71082 in __libc_start_main /build/glibc-SzIz7B/glibc-2.31/csu/../csu/libc-start.c:308:16
  ◦ #14 0x426cbd in _start (/load_fuzzer+0x426cbd)
▶ AsanReport
▼ Source
  ◦     719
  ◦     720         while (current >= 0)
  ◦     721         {
  ◦     722             chain.push_back(current);
  ◦ --->723             current = table[static_cast<std::size_t>(current)];
  ◦     724         }
  ◦     725
  ◦     726         return chain;
  ◦     727     }
  ◦     728
                                                                                                   Press q to exit
```

casr-afl: интеграция CASR с AFL++

ИСП РАН

`casr-afl` from CASR tools provides comfortable triaging for crashes found by AFL++. Reports are clustered and contain severity and other information.

```
casr-afl -i /path/to/afl/out/dir -o /path/to/casr/out/dir
```

Из документации по AFL++

```
root@splash:/fuzz# casr-afl -i /casr/tests/casr_tests/bin/afl-out-xlnt -o casr-out
11:09:26 [INFO] Generating CASR reports...
11:09:34 [WARN] casr-gdb: no crash on input /casr/tests/casr_tests/bin/afl-out-xlnt/afl_s01-worker/crashes/id:000012,sig:00,sync:afl_main-worker,src:000730
11:09:46 [WARN] casr-gdb: no crash on input /casr/tests/casr_tests/bin/afl-out-xlnt/afl_s01-worker/crashes/id:000024,sig:00,src:000772,time:9531749,execs:109316,op:havoc,rep:4
11:09:53 [WARN] casr-gdb: no crash on input /casr/tests/casr_tests/bin/afl-out-xlnt/afl_s01-worker/crashes/id:000021,sig:00,src:000272,time:5832328,execs:69251,op:havoc,rep:8
11:09:53 [WARN] casr-gdb: no crash on input /casr/tests/casr_tests/bin/afl-out-xlnt/afl_s01-worker/crashes/id:000022,sig:00,src:000272,time:5834254,execs:69261,op:havoc,rep:4
11:09:54 [WARN] casr-gdb: no crash on input /casr/tests/casr_tests/bin/afl-out-xlnt/afl_s01-worker/crashes/id:000020,sig:00,src:000018,time:5281022,execs:51619,op:havoc,rep:8
11:10:11 [INFO] Deduplicating CASR reports...
11:10:11 [INFO] Number of reports before deduplication: 54. Number of reports after deduplication: 33
11:10:11 [INFO] Clustering CASR reports...
11:10:12 [INFO] Number of clusters: 21
root@splash:/fuzz#
```

Результат работы casr-afl

Планы по развитию

- Гибридный фаззинг и анализ аварийных завершений для AARCH64(Baikal-M)/RISC-V
- Фаззинг и анализ аварийных завершений для программ на Python и Java