Аверьянов П. Д.

ББМО-01-24

Практическое задание 7

Фаззинг-тестирование

Задачи работы:

1. Необходимо подготовить виртуальную машину с ОС Linux

2. Запустить ВМ и установить в нее Docker

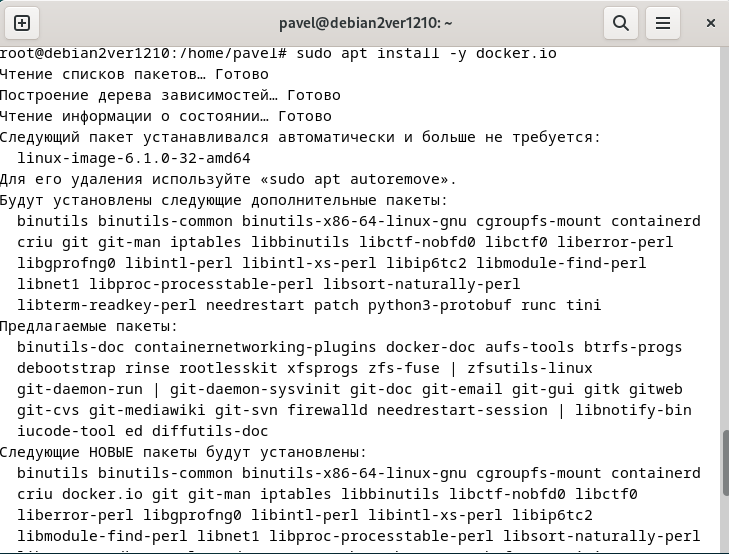
3. Выполнить команду `docker pull aflplusplus/aflplusplus

4. Выбрать фаззинг-цель для тестирования (простое приложение на `C`) и запустить фаззинг-тестирование исследуемой программы.

5. Подготовить отчет со скриншотами и командами, выполненными внутри контейнера.

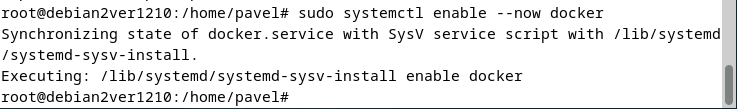
**Фаззинг** (или **fuzzing**) — это метод тестирования программного обеспечения, который используется для обнаружения уязвимостей и ошибок в приложениях. Этот подход включает в себя автоматическую генерацию и отправку случайных или некорректных данных (входных данных) в программу с целью выявления сбоев,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

**-y**: Этот флаг автоматически отвечает "да" на все запросы, которые могут возникнуть во время установки. Это позволяет избежать необходимости вручную подтверждать установку, что удобно для автоматизации процесса.





**--now**: Этот флаг указывает, что служба должна быть запущена немедленно, а не только включена для автоматического запуска при следующей загрузке. Это означает, что команда не только активирует службу, но и запускает ее сразу.

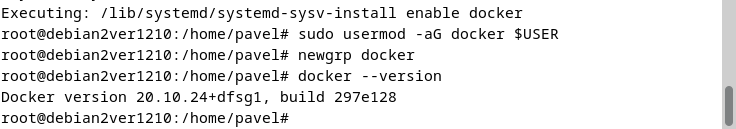


**-a**: Этот флаг означает "append" (добавить). Он используется для добавления пользователя в указанную группу, не удаляя его из других групп, к которым он уже принадлежит. Без этого флага пользователь был бы удален из всех групп, кроме указанной.

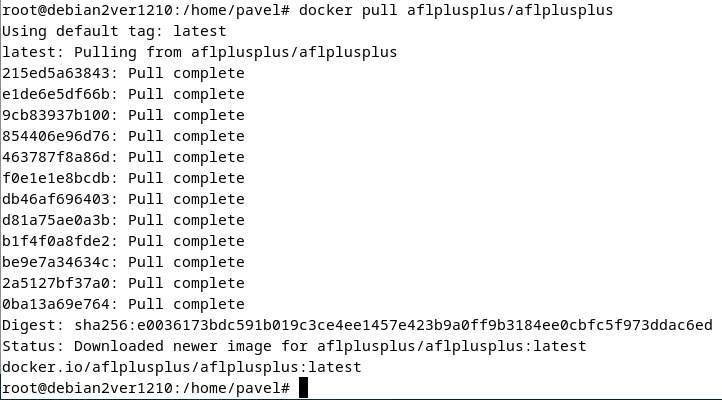
**-G**: Этот флаг указывает, что следующая часть команды будет содержать список групп, в которые нужно добавить пользователя. В данном случае это группа docker.



newgrp docker # Активация изменений без перезагрузки



Команда docker pull aflplusplus/aflplusplus используется для загрузки Docker-образа из Docker Hub, который содержит инструмент для фаззинга под названием AFL++.



Docker создает и запускает контейнер на основе образа AFL++

**-t**: Создает псевдотерминал (tty), что позволяет взаимодействовать с контейнером через терминал.

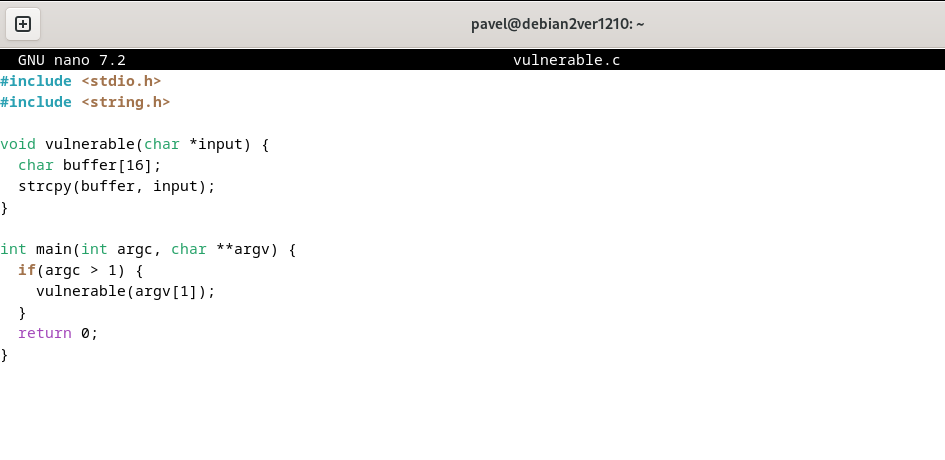
**-i**: Позволяет взаимодействовать с контейнером в интерактивном режиме, сохраняя стандартный ввод (stdin) открытым.

**~/aflt**: Это путь к локальной директории на вашем компьютере, где могут находиться исходные файлы или данные, которые вы хотите использовать для фаззинга.

**/src**: Это путь внутри контейнера, куда будет смонтирована локальная директория. Таким образом, все файлы из ~/aflt будут доступны в контейнере по пути /src.



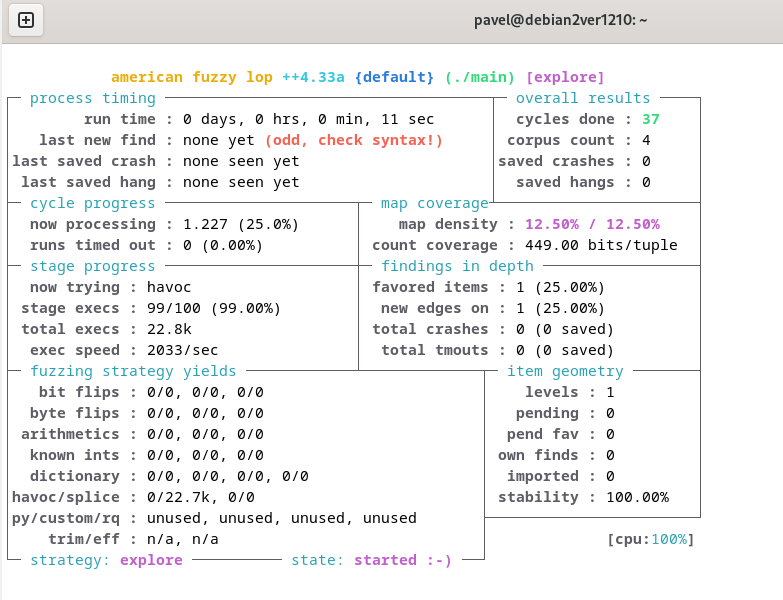
Далее напишу небольшой код на C для запуска фазззинга

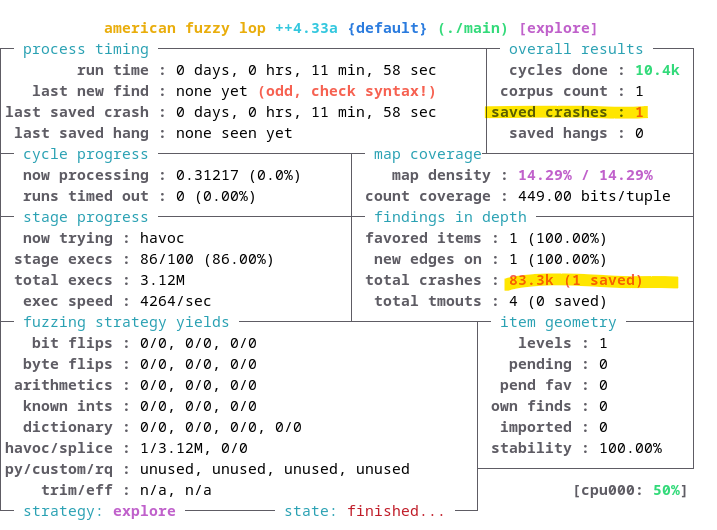


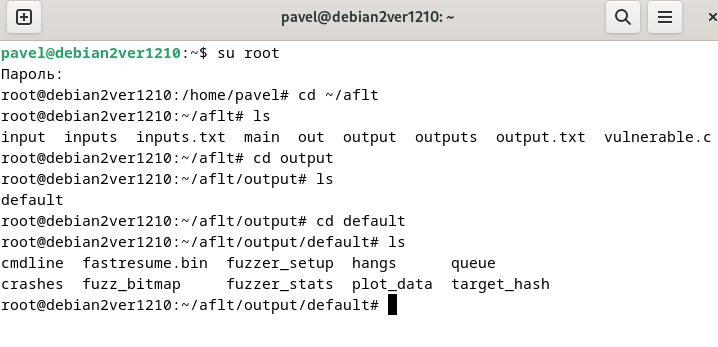
И внесу измнения в входной файл

**afl-fuzz**: Это основной исполняемый файл инструмента AFL++, который отвечает за фаззинг. Он генерирует случайные или мутационные входные данные и отправляет их в целевую программу для тестирования.









**Описание компонентов фаззера**

1. **Очередь (queue)**: Это место, где хранятся входные данные, используемые фаззером для создания новых тестовых вариантов.
2. **Сбои (crashes)**: Каталог, в который сохраняются файлы, вызвавшие сбои программы во время тестирования.
3. **Зависания (hangs)**: Папка, где хранятся тестовые файлы, которые привели к зависанию программы, то есть к длительному выполнению без завершения.
4. **Статистика фаззера (fuzzer\_stats)**: Файлы, содержащие статистические данные о работе фаззера, такие как количество выполненных циклов и покрытие кода.
5. **Данные для графиков (plot\_data)**: Информация, используемая для построения графиков, например, зависимость покрытия кода от времени.
6. **Хэш целевых файлов (target\_hash)**: Данные о хэшировании целевых файлов, которые помогают отслеживать уникальность входных данных.
7. **История команд (cmdline)**: Запись команд, использованных для запуска фаззера, включая аргументы командной строки.
8. **Битмап покрытия (fuzz\_bitmap)**: Данные, которые отслеживают покрытие кода, включая битмапы, используемые для определения новых ветвей выполнения.
9. **Настройки фаззера (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.