BÁO CÁO TỔNG KẾT ĐỒ ÁN MÔN HỌC

**Môn học: Lập trình an toàn và khai thác lỗ hổng phần mềm**

**Tên chủ đề: Binary-level Directed Fuzzing for Use-After-Free Vulnerabilities**

*Mã nhóm: Nhóm 03 Mã đề tài: CK14*

**Lớp**: **NT521.O11.ANTT**

1. **THÔNG TIN THÀNH VIÊN NHÓM:**

*(Sinh viên liệt kê tất cả các thành viên trong nhóm)*

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Họ và tên** | **MSSV** | **Email** |
| 1 | Nguyễn Ngọc Trà My | 21520353 | 21520353@gm.uit.edu.vn |
| 2 | Bùi Hoàng Trúc Anh | 21521817 | 21521817@gm.uit.edu.vn |
| 3 | Lê Hoàng Oanh | 21521253 | [21521253@gm.uit.edu.vn](mailto:21521253@gm.uit.edu.vn) |
| 4 | Huỳnh Minh Tân Tiến | 21521520 | 21521520@gm.uit.edu.vn |

1. **TÓM TẮT NỘI DUNG THỰC HIỆN:[[1]](#footnote-2)**

*Phần này tóm tắt nội dung của đồ án, sinh viên báo cáo nội dung chi tiết ở phần BÁO CÁO CHI TIẾT*

* 1. **Chủ đề nghiên cứu trong lĩnh vực An toàn phần mềm:**

☒ Phát hiện lỗ hổng bảo mật phần mềm

¨ Khai thác lỗ hổng bảo mật phần mềm

¨ Sửa lỗi bảo mật phần mềm tự động

¨ Lập trình an toàn

¨ Khác: ……………….

* 1. **Tên bài báo tham khảo chính:**

|  |
| --- |
| Tiếng Anh: M.-D. Nguyen, S. Bardin, R. Bonichon, R. Groz, and M. Lemerre, “Binary-level Directed Fuzzing for Use-After-Free Vulnerabilities.” Accessed: Oct. 29, 2023. [Online].  Tiếng Việt: Kiểm thử Fuzzing có định hướng ở mức độ nhị phân cho các lỗ hổng Use-After-Free |

* 1. **Tên bài báo tham khảo khác (nếu có):**

|  |
| --- |
| <liệt kê 1 số bài báo có liên quan/tương tự hoặc có kỹ thuật mà nhóm sử dụng khi thực hiện đồ án (vd khi thực nghiệm...) (nếu có)> |

* 1. **Tóm tắt nội dung chính:**

|  |
| --- |
| UAFuzz là một DGF được tùy chỉnh để tập trung tìm kiếm lỗi ở mức độ nhị phân liên quan đến UAF (use after free), lỗ hổng liên quan đến việc các phần tử heap được sử dụng sau khi giải phóng, gây ra nguy cơ đối với dữ liệu.  UAFuzz phát hiện ra các UAF bằng cách tạo và chọn lựa các đầu vào có các sự kiện cấp phát, sử dụng và giải phóng, để làm cho chương trình lỗi và ghi nhận lại các bug traces.  Phương pháp mà UAFuzz tiếp cận: đề xuất ra các số liệu để đánh giá độ phát hiện ra UAF; tạo ra các thuật toán để đánh giá và lựa chọn input: seed selection, power schedule, cut-ede coverage metric, energy assignment, ... và sử dụng các số liệu có sẵn để xử lý và phân loại lỗi.  Một số thuật toán mà UAF sử dụng: thuật toán Seed Selection, thuật toán tính toán seed distance, thuật toán tính toán độ bao phủ của seed, thuật toán xác định cạnh cắt và cạnh không cắt nhằm mục tiêu cải thiện hiệu suất và kết quả fuzzing.  Triển khai UAFuzz: Đầu vào của hệ thống tổng thể bao gồm một tập các "seed" ban đầu, mã nguồn mục tiêu tại dạng nhị phân và các vị trí mục tiêu được trích xuất từ dấu vết lỗi. Đầu ra là một tập các đầu vào gây ra lỗi duy nhất. Bản nguyên mẫu được xây dựng trên cơ sở của AFL 2.52b và QEMU 2.10.0 để thực hiện fuzzing, và nền tảng phân tích nhị phân BINSEC để thực hiện phân tích tĩnh (nhẹ).  Đánh giá UAFuzz: đánh giá dựa trên 4 câu hỏi: khả năng tái tạo lỗi, chi phí, phân loại UAF, và những đóng góp của công cụ để thấy được sự vượt trội của UAFuzz so với các fuzzer khac như Hawkeye, AFLGo, ...  Kết luận: UAFuzz là một DGB có hiệu năng vượt trội về việc tìm kiếm các lỗ hổng UAF so với các fuzzer đối thủ |

* 1. **Công việc/tính năng/kỹ thuật mà nhóm thực hiện lập trình và triển khai cho demo:**

|  |
| --- |
| <liệt kê các công việc mà nhóm thực hiện cho đề tài dựa trên phân tích phương pháp/hệ thống được sử dụng trong bài báo đã tham khảo>  +Cài đặt môi trường: OS: ubuntu, các packet cần thiết: binsec, ida for interface +Cài đặt fuzzer: AFL +Fuzzing thử nghiệm với code lỗi UAF +Tùy biến AFL thành Uafuzz  + Fuzzing thực nghiệm lại  + Đánh giá  <liệt kê các công việc đã thực hiện+ tóm tắt kết quả của công việc này>  +Cài đặt môi trường: OS: ubuntu, các thư viện hỗ trợ +Cài đặt fuzzer AFL +Fuzzing thử nghiệm với code mẫu.  +Tùy biến AFL thành UAFUZZ  + Fuzzing thực nghiệm lại : fuzzing trên file code của bài báo không thành công. |

1. **TỰ ĐÁNH GIÁ MỨC ĐỘ HOÀN THÀNH SO VỚI KẾ HOẠCH THỰC HIỆN:**

|  |
| --- |
| 85% |

1. **NHẬT KÝ PHÂN CÔNG NHIỆM VỤ:**

|  |  |  |
| --- | --- | --- |
| **STT** | **Công việc** | **Phân công nhiệm vụ** |
| 1 | Tìm hiểu lý thuyết | <Huỳnh Minh Tân Tiến, Bùi Hoàng Trúc Anh, Lê Hoàng Oanh, Nguyễn Ngọc Trà My> |
| 2 | Viết báo cáo | <Huỳnh Minh Tân Tiến, Bùi Hoàng Trúc Anh, Lê Hoàng Oanh> |
| 3 | Demo | <Bùi Hoàng Trúc Anh, Lê Hoàng Oanh, Nguyễn Ngọc Trà My> |
| 4 | Làm slide | <Huỳnh Minh Tân Tiến, Bùi Hoàng Trúc Anh> |
| 5 | Thuyết trình | <Nguyễn Ngọc Trà My> |

BÁO CÁO TỔNG KẾT CHI TIẾT

**Phần bên dưới của báo cáo này là tài liệu báo cáo tổng kết - chi tiết của nhóm thực hiện cho đề tài này.**

*Qui định: Mô tả các bước thực hiện/ Phương pháp thực hiện/Nội dung tìm hiểu (Ảnh chụp màn hình, số liệu thống kê trong bảng biểu, có giải thích)*

# GIỚI THIỆU TỔNG QUAN

*<Trình bày tổng quan vấn đề mà bài báo muốn giải quyết/lý do phương pháp được đề xuất>*

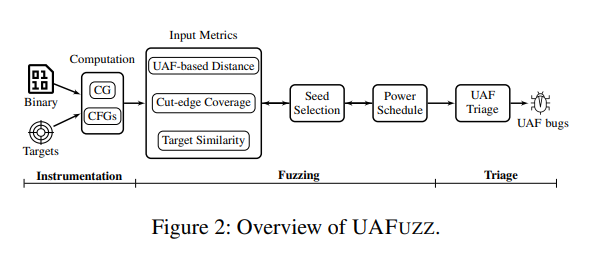
Các phần mềm kiểm thử hiện tại không được thiết kế để tối ưu việc phát hiện lỗi UAF (Use-After-Free) do đó để tăng cường bảo mật, UAFuzz được tạo ra để kiểm thử UAF ở binary-level.

# PHƯƠNG PHÁP THỰC HIỆN

## Phương pháp được đề xuất

*<Trình bày chi tiết kiến trúc, thành phần của hệ thống trong bài báo>*

Về kiến trúc của hệ thống

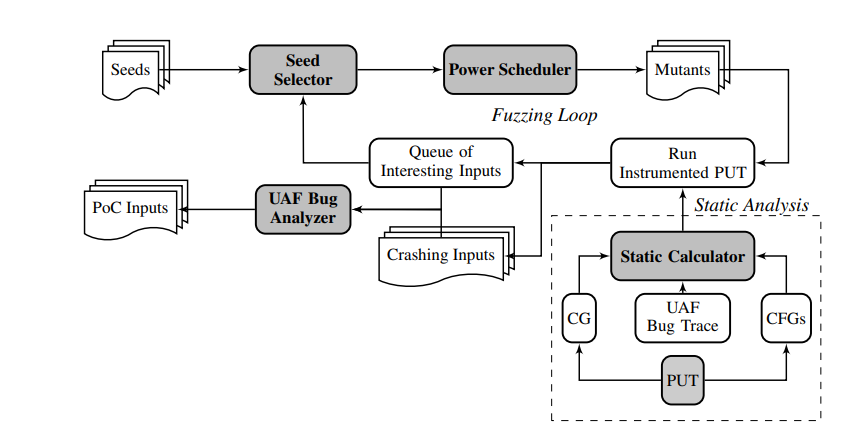


Uafuzz: có các thành phần:  
 - Instrumentation: chịu trách nhiệm cho việc phân tích tĩnh trước mục tiêu  
 - Fuzzing:  
 + Seed Selection: xây dựng thuật toán để chọn đầu vào cho fuzzing để tìm ra đầu vào có độ bao phủ với mức lớn nhất   
 + Power Schedule: xây dụng một power schedule mới dựa trên việc cho rằng nên phân bổ nhiều năng lượng hơn cho hạt giống trong những trường hợp sau:

• các seed gần nhau hơn (sử dụng khoảng cách seed)   
 • các seed giống với mục tiêu hơn   
 • seed đưa ra quyết định tốt hơn tại các điểm nối mã quan trọng.  
 + Input Metrics:  
 • UAF-based distances: Điều chỉnh khoảng cách giữa hoạt động trong biểu đồ cuộc gọi để ưu tiên các đường đi tuần tự thông qua ba sự kiện UAF alloc, free và use. Điều này được thực hiện bằng cách giảm trọng lượng của các cạnh trong biểu đồ cuộc gọi có khả năng nằm giữa các sự kiện này, sử dụng phân tích tĩnh nhẹ.  
 • Cut-edge Coverage Metric: Xây dựng thuật toán số liệu đạt được điểm trung gian giữa hai điều đã nói ở trên bằng cách đo lường tiến trình ở cấp biên nhưng chỉ trên các nút quyết định quan trọng.   
 • Target Similarity Metric: Xây dựng các 4 chỉ số (Target prefix, UAF prefix, Target bag, UAF bag) để đánh giá sự tương đồng - độ chính xác của việc thực thi các seed với UAF bug trace.

- Triage: Phân loại các lỗi đã phát hiện dựa trên các thông tin sẵn, xác định các trước các seed có khả năng gây lỗi theo 3 sự kiện của lỗi UAF: alloc – free – use

Cách UAFuzz hoạt động



1. DGF thực hiện trước tiên một phân tích tĩnh (ví dụ, tính toán khoảng cách đến mục tiêu cho mỗi khối cơ bản) và chèn các lệnh để thu thập thông tin về phủ động hoặc khoảng cách động
2. Công cụ tạo ra các đầu vào mới bằng cách biến đổi lặp đi lặp lại các đầu vào được chọn từ hàng đợi kiểm thử S cho đến khi hết thời gian chờ đợi. Một đầu vào được chọn nếu nó được ưa chuộng (tức là, được tin là đáng chú ý) hoặc với xác suất nhỏ α Sau đó, DGF gán năng lượng (còn được gọi là số lượng biến đổi M cần tạo) cho seed được chọn s. Sau đó, công cụ tạo ra M đầu vào mới bằng cách ngẫu nhiên áp dụng một số toán tử biến đổi được xác định trước trên seed s và theo dõi việc thực thi của chúng. Nếu mutant được tạo ra s0 làm cho chương trình bị crash, nó sẽ được thêm vào tập S0 của các đầu vào làm cho chương trình crash. Ngoài ra, các mutant mới được thêm vào hàng đợi kiểm thử;
3. DGF trả về S0 là tập hợp các đầu vào gây lỗi.

## Nội dung cải tiến/phát triển thêm (nếu có)

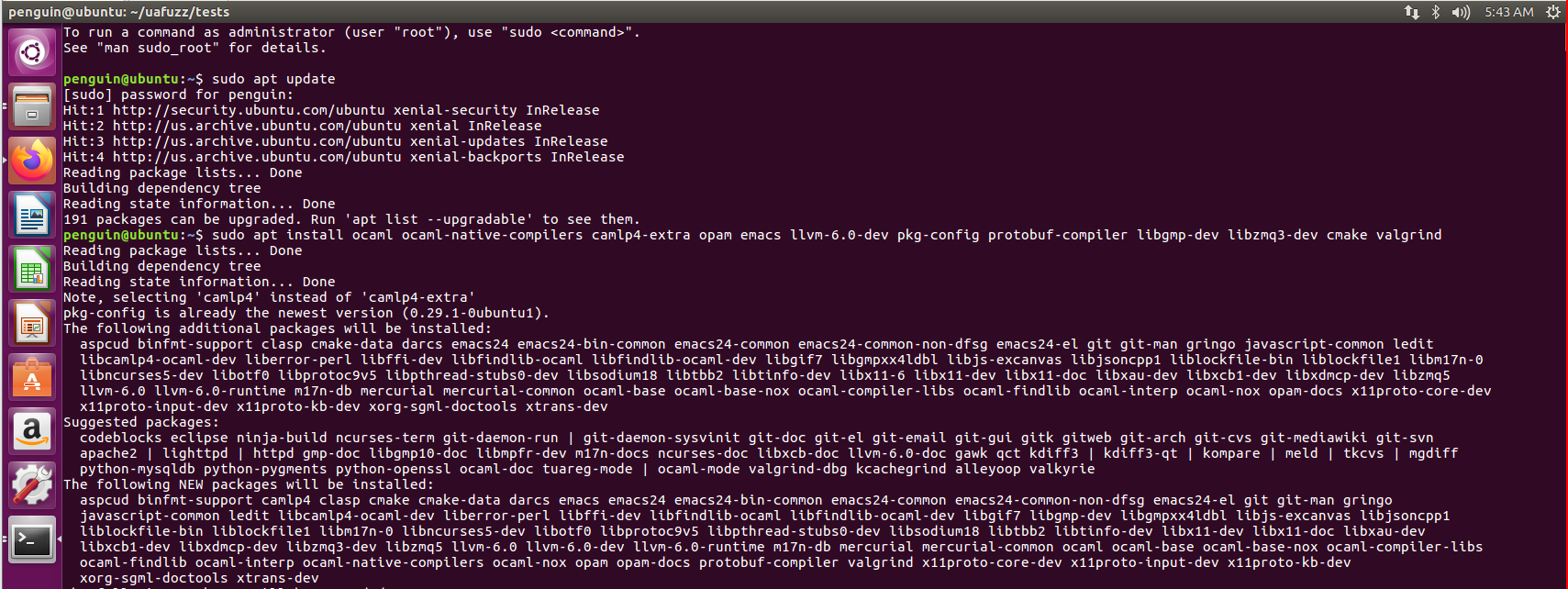
*<Trình bày nội dung/kỹ thuật mà nhóm thực hiện thêm so với phương pháp đề xuất của bài báo: Lý do? Chi tiết nội dung?>*

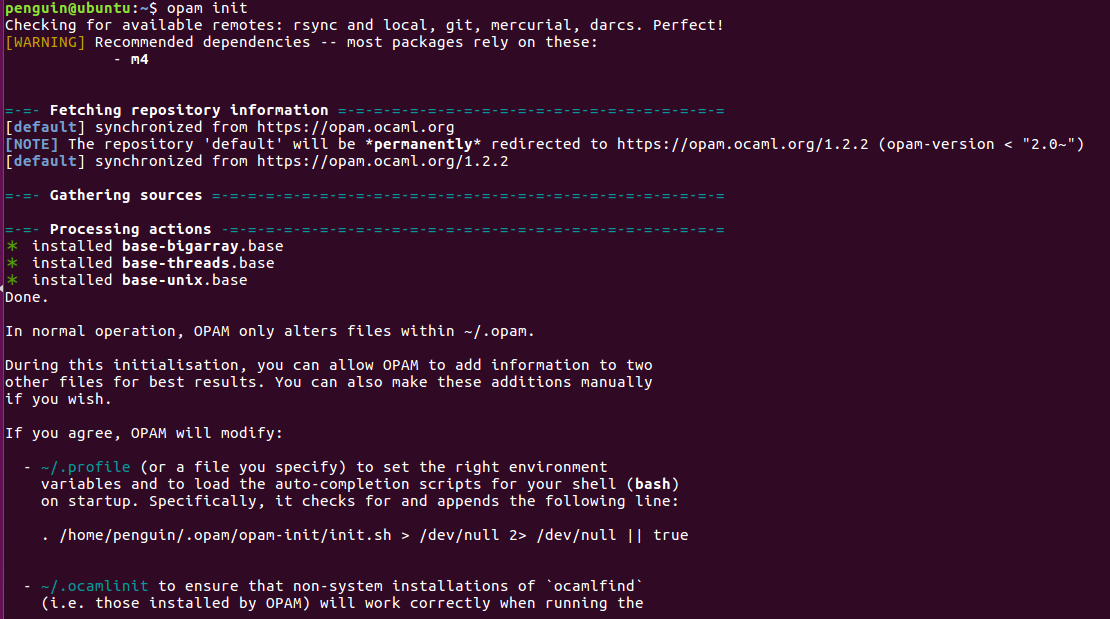
# CHI TIẾT HIỆN THỰC VÀ THỰC NGHIỆM PHƯƠNG PHÁP

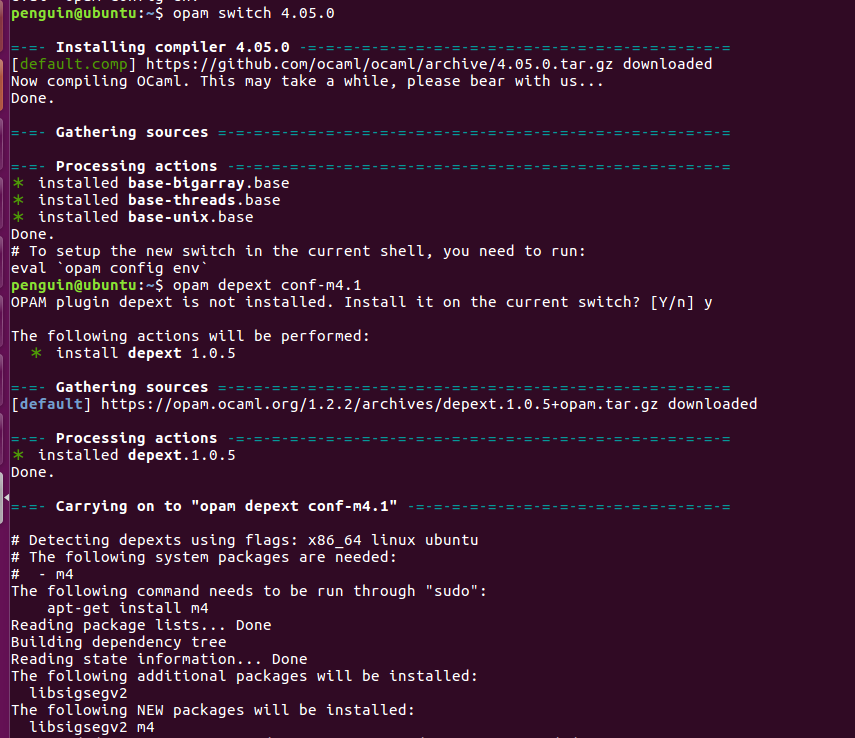
## Hiện thực phương pháp

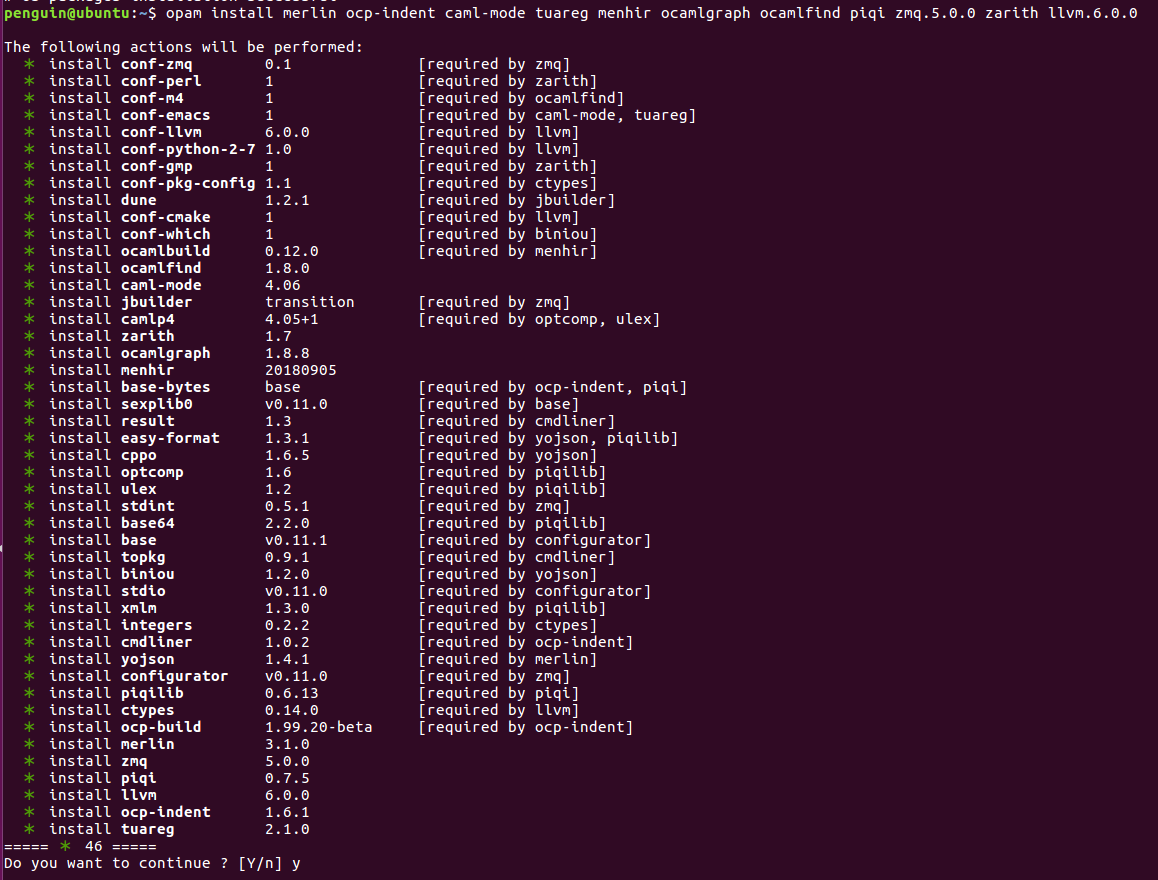
*<Cách cài đặt, lập trình trên máy tính, cấu hình máy tính sử dụng, chuẩn bị dữ liệu, v.v>*

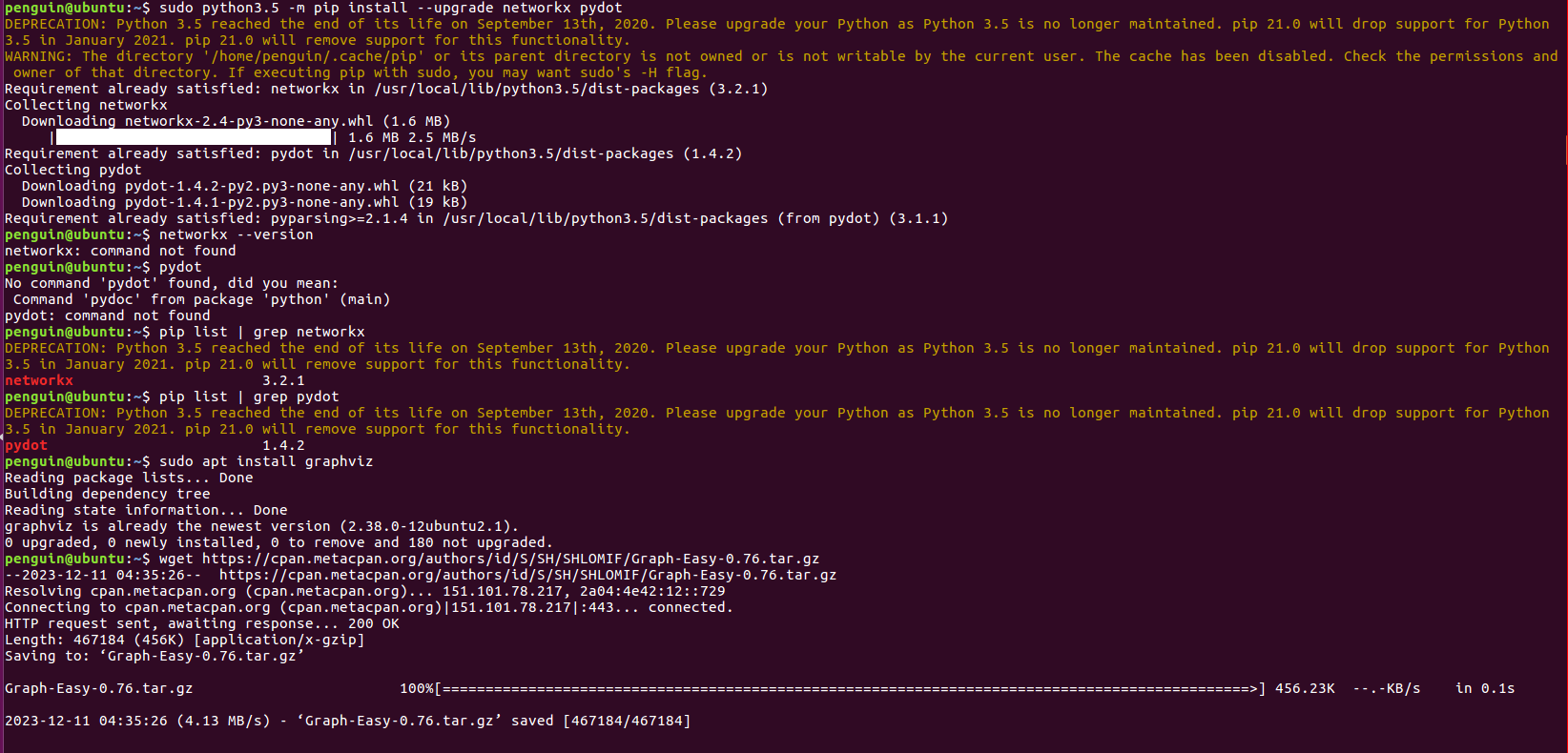
*Nhóm sinh viên báo cáo các nội dung mà nhóm đã thực hiện, có thể là 1 phần hoặc toàn bộ nội dung của bài báo. Nếu nội dung thực hiện có khác biệt với bài báo (như cấu hình, tập dữ liệu, kết quả,...), sinh viên cần chỉ rõ khác biệt đó và nguyên nhân.*

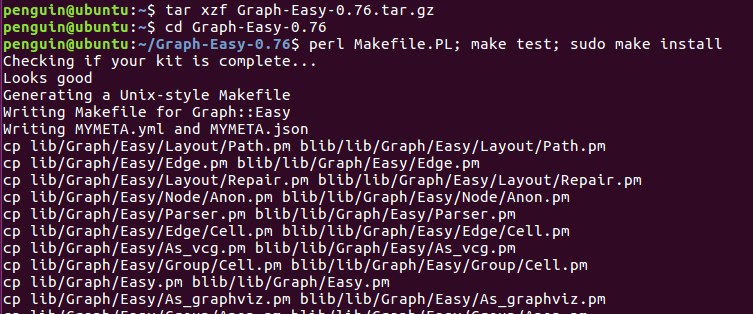
Cài đặt các thành phần cần thiết theo hướng dẫn của bài báo:  


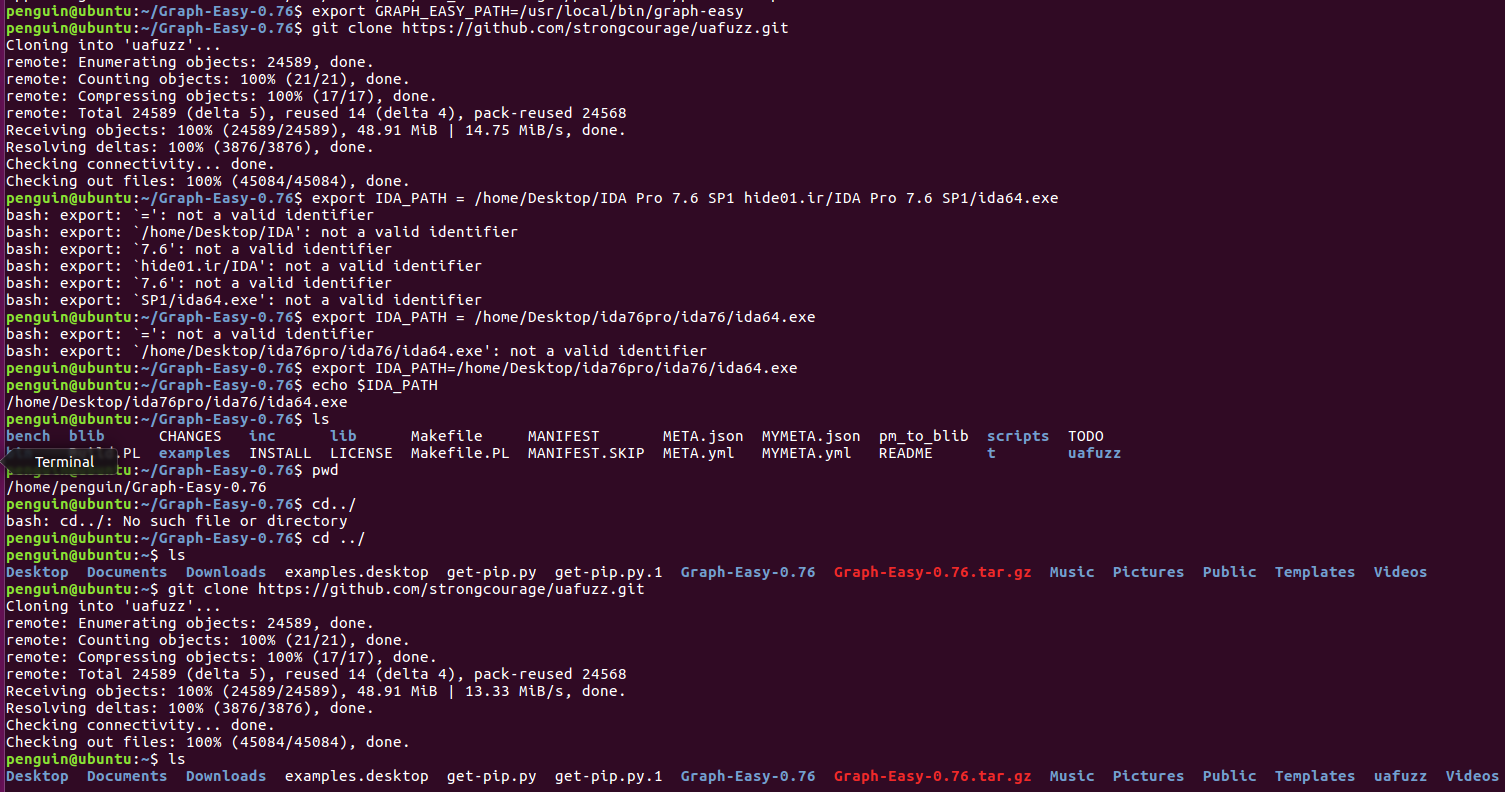






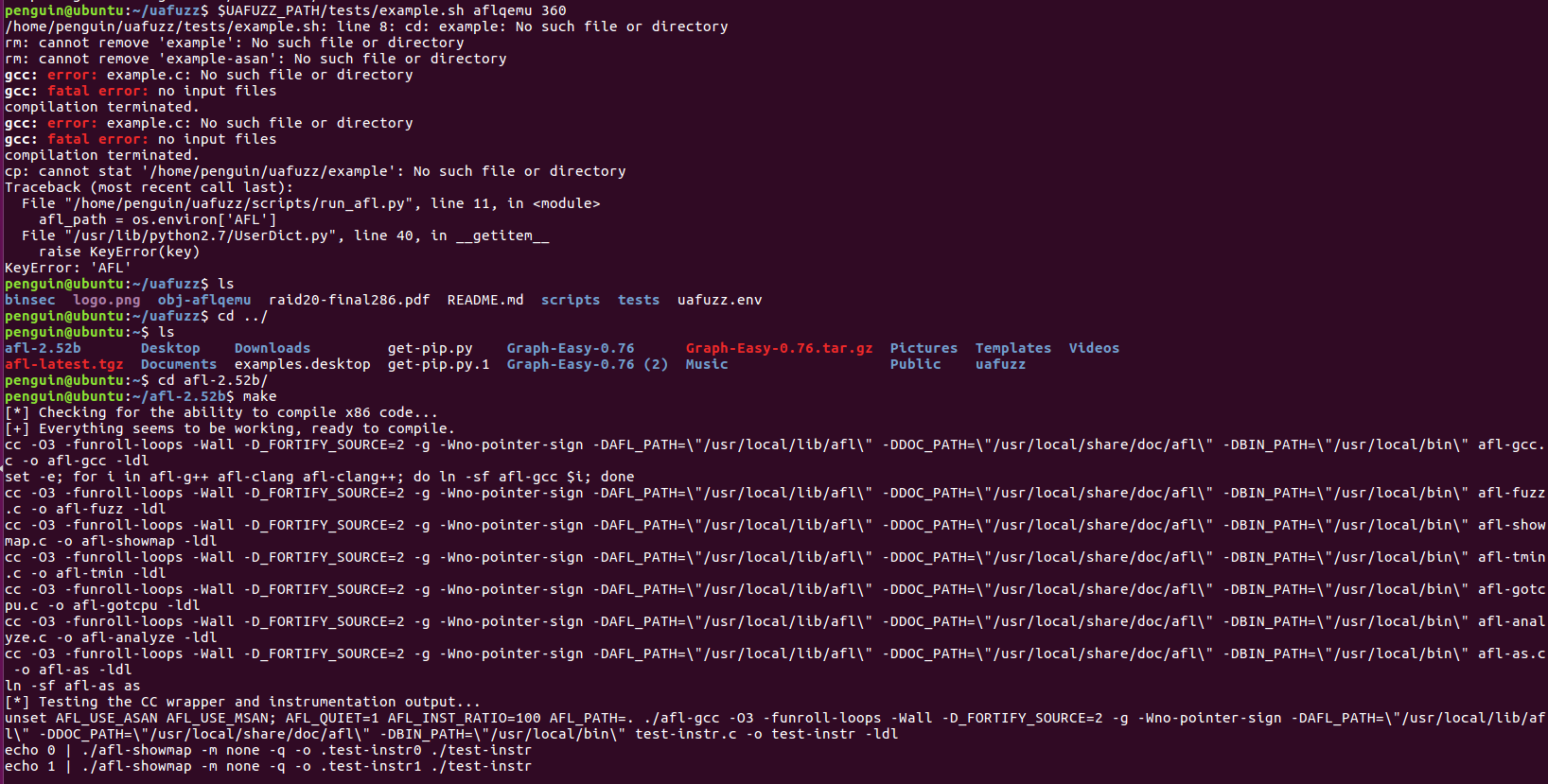


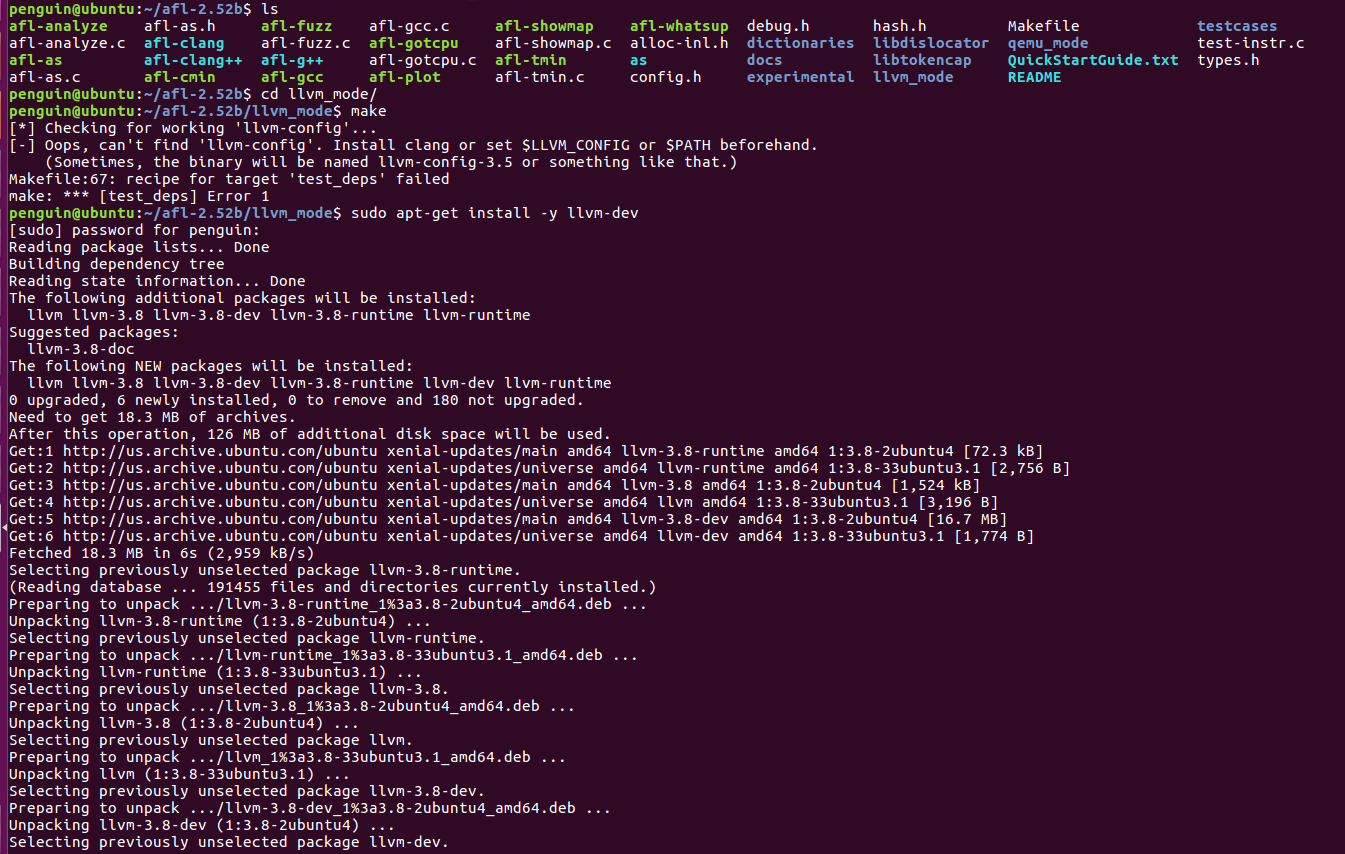


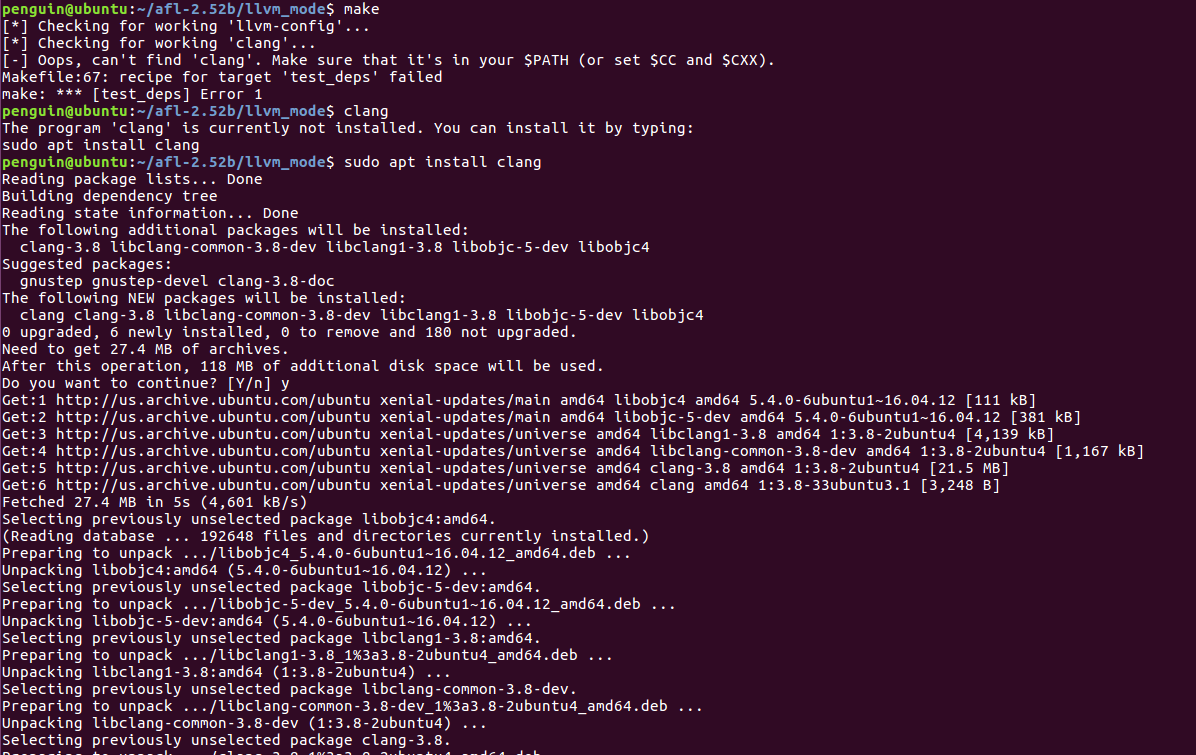


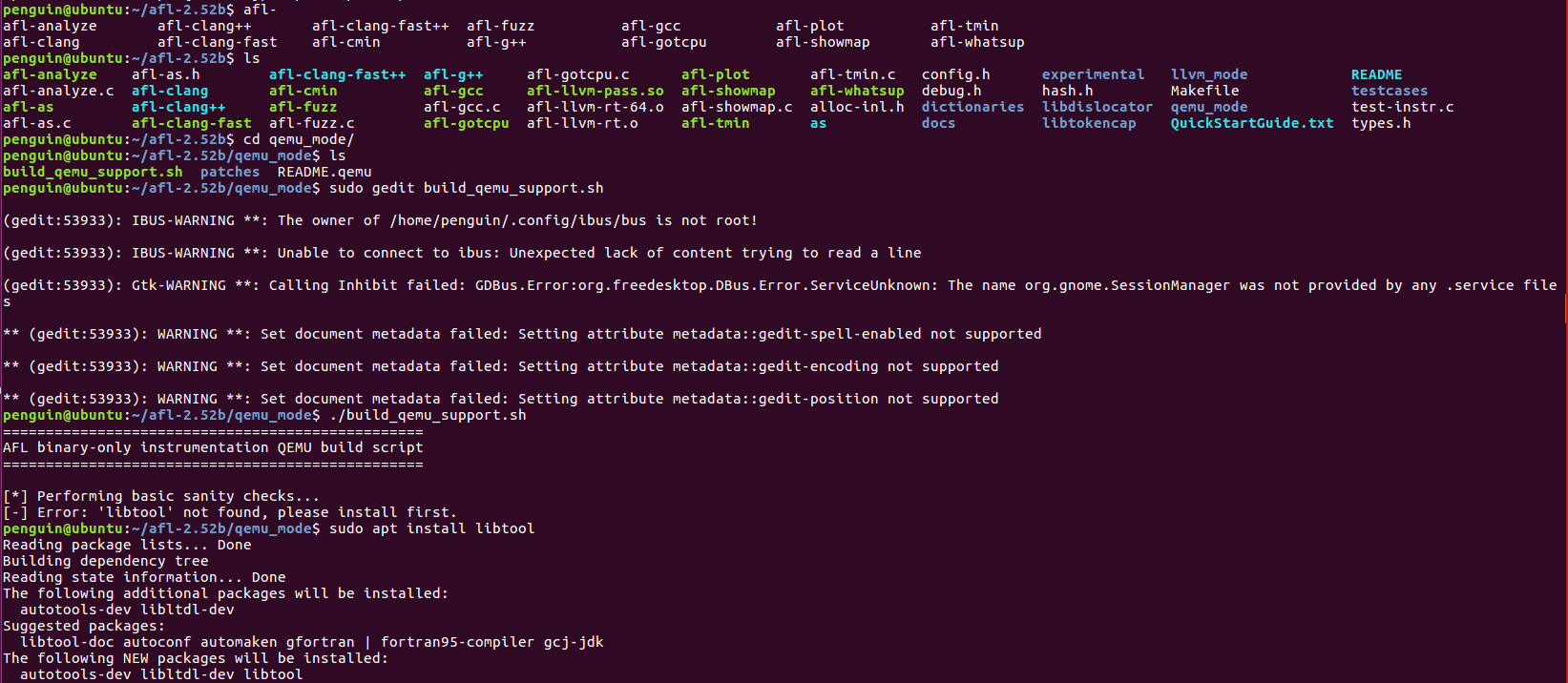


Tải AFL v2.52b

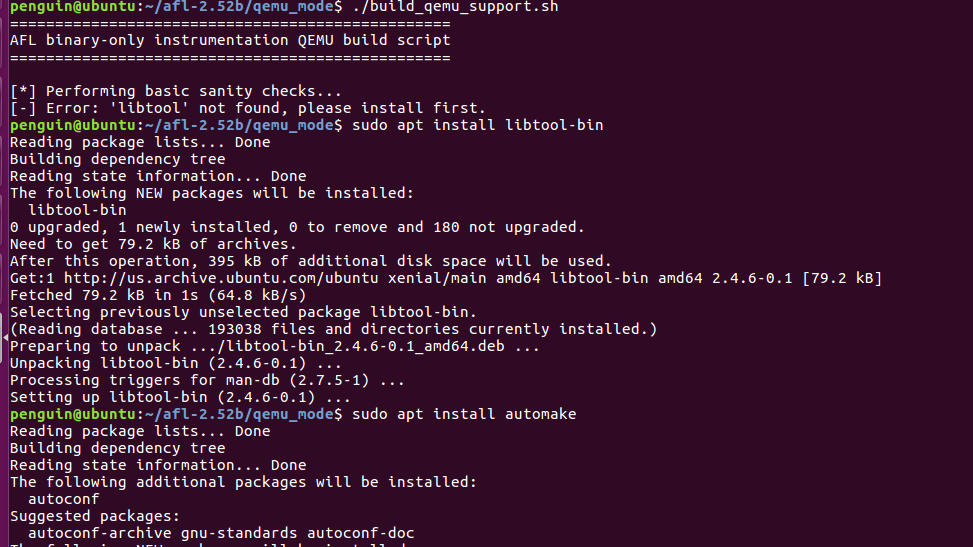


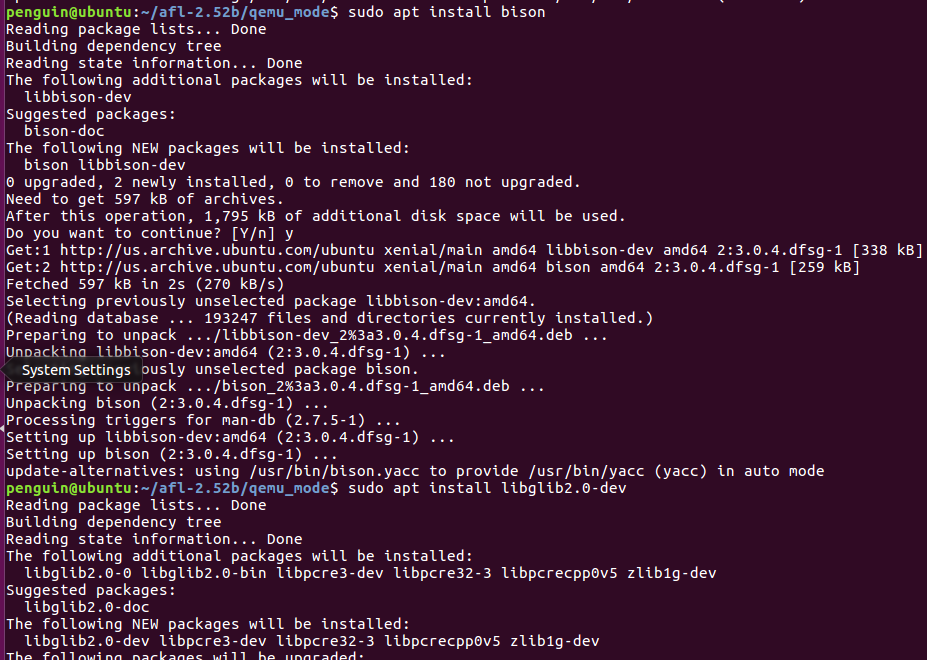


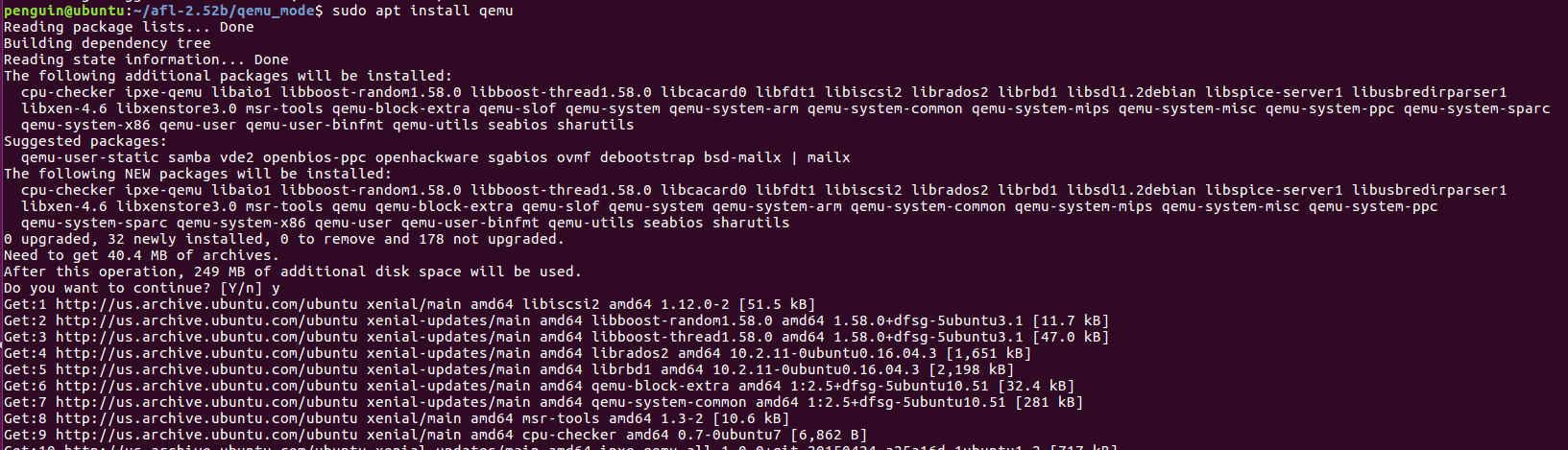


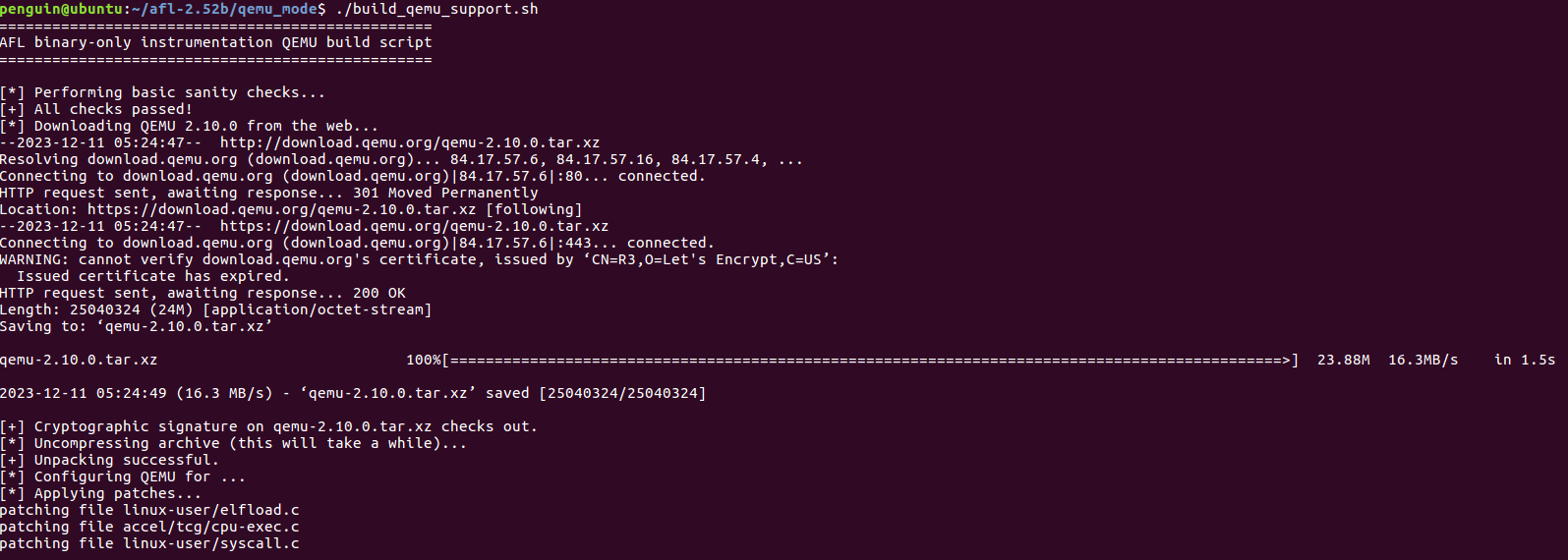


Tải những gói cần thiết để cài đặt qemu mode trong afl-2.52b









## Thực nghiệm phương pháp

### Kịch bản thực nghiệm

<*Mô tả các kịch bản mà nhóm sinh viên thực hiện cho phần thực nghiệm, có thể là toàn bộ hoặc 1 phần trong các kịch bản của bài báo hoặc kịch bản mới do sinh viên đề xuất (ghi chú rõ ràng)>*

Dùng công cụ AFL để scan thử lỗi UAF trên file example.c   
File code example.c theo bài báo.   
A screenshot of a computer program

Description automatically generated  
Thực hiện fuzzing trên file:   
Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Description automatically generated

Công cụ tốn nhiều thời gian để scan nhưng vẫn không thể scan được lỗi UAF

Dùng công cụ UAFUZZ đã xây dựng được để scan lỗi UAF trên file example.c

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ

Description automatically generated

Công cụ UAFUZZ vẫn còn một vài chỗ chưa hoàn thiện, dẫn đến kết quả cuối cùng là dù đã tùy biến để tìm kiếm các lỗ hổng UAF, nhưng vẫn không scan thấy được các bug.

### Kết quả thực nghiệm

*<mô tả hình ảnh kết quả thực nghiệm, bảng biểu số liệu thống kê từ thực nghiệm,* ***CẦN NHẬN XÉT*** *về kết quả thu được.>*

# HƯỚNG PHÁT TRIỂN

*<Nêu hướng phát triển tiềm năng của đề tài này trong tương lai. Nhận xét về tính ứng dụng của đề tài>.*

---

***Sinh viên đọc kỹ yêu cầu trình bày bên dưới trang này***

# YÊU CẦU CHUNG

* Sinh viên tìm hiểu và thực hiện bài tập theo yêu cầu, hướng dẫn.
* Nộp báo cáo kết quả chi tiết những việc (**Report**) bạn đã thực hiện, quan sát thấy và kèm ảnh chụp màn hình kết quả (nếu có); giải thích cho quan sát (nếu có).
* Sinh viên báo cáo kết quả thực hiện và nộp bài.

**Báo cáo:**

* File .PDF. Tập trung vào nội dung, không mô tả lý thuyết.
* Đặt tên theo định dạng: [Mã lớp]-Project\_Final\_NhomX\_Madetai. (trong đó X và Madetai là mã số thứ tự nhóm và Mã đề tài trong danh sách đăng ký nhóm đồ án).

*Ví dụ: [NT521.O11.ANTT]-Project\_Final\_Nhom03\_CK01.*

* Báo cáo tổng kết được nộp cùng slide trình bày và source code trên moodle, nhóm sinh viên tạo thư mục theo quy ước cho trước trên moodle khi nộp các file.
* Nộp file báo cáo trên theo thời gian đã thống nhất tại courses.uit.edu.vn.

**Đánh giá**:

* Hoàn thành tốt yêu cầu được giao.
* Có nội dung mở rộng, ứng dụng.

*Bài sao chép, trễ, … sẽ được xử lý tùy mức độ vi phạm.*

**HẾT**

1. Ghi nội dung tương ứng theo mô tả [↑](#footnote-ref-2)