ICDF

- Các công cụ fuzz testing hiện tại ( điển hình là *xFuzz* và *ILF* giảm số lượng tổ hợp chuỗi giao dịch cần thực thi bằng mô hình học máy) thường bỏ qua việc ưu tiên các hàm liên quan đến biến trạng thái trong quá trình tạo chuỗi giao dịch. Điều này dẫn đến tập dữ liệu không cân bằng gây lãng phí tài nguyên và giảm hiệu quả kiểm tra của mô hình

- Phân tích luồng dữ liệu liên hợp đồng (ICDF) là bước quan trọng trong CrossFuzz nhằm tối ưu hóa chiến lược biến đổi chuỗi giao dịch, giúp cải thiện khả năng phát hiện lỗ hổng. CrossFuzz sẽ tiếp cận phân tích ICDF theo hướng xem xét cách các biến trạng thái được định nghĩa, sử dụng trong các hợp đồng riêng lẻ và quan hệ giữa các hàm trong các hợp đồng khác nhau

- Cụ thể thì đầu tiên CrossFuzz sẽ phân tích quan hệ gọi hàm giữa các hợp đồng ( hàm này có thể gọi hàm của hợp đồng khác ). Hành động này nhằm thu được tên của các hợp đồng và hàm được gọi. Lưu ý là nếu 1 hàm áp dụng sửa đổi/ kiểm tra cho hàm kia thì thì hàm áp dụng sửa đổi sẽ được thực thi trước cho chính hàm kia

- Tiếp theo các biến trạng thái mà các hàm xác định và sử dụng sẽ được ghi lại trong define\_map và use\_map

+ define\_map: Ghi lại các biến trạng thái được định nghĩa (thay đổi giá trị) bởi các hàm

+ use\_map: Ghi lại các biến trạng thái được sử dụng (đọc giá trị) bởi các hàm trong hợp đồng thông minh

-> Từ define-map và use\_map thì CrossFuzz sẽ ưu tiên các giao dịch liên quan đến biến trạng thái chưa được khởi tạo trong các giao dịch trước. Điều này giúp giảm số lượng giao dịch không cần thiết và tăng hiệu quả kiểm thử

A diagram of a business process

Description automatically generated

- Trong hợp đồng PM:

+ addAddress: cập nhật danh sách địa chỉ được phép

+ isPermitted: Kiểm tra xem địa chỉ (msg.sender) có nằm trong danh sách permittedAddresses không

- Trong hợp đồng Hold:

+ release: Gọi hàm isPermitted từ **PermissionManager** để kiểm tra quyền trước khi thực hiện logic phát hành Ether

+ getbalance: Hàm này có thể lấy thông tin số dư Ether đã được phát hành, có thể liên quan đến trạng thái của hợp đồng

- define: addAddress định nghĩa biến permittedAddresses, changeStage cập nhật trạng thái của hợp đồng

- use: isPermitted sử dụng biến permittedAddresses, release sử dụng hàm isPermitted, getbalance

Xác định vấn đề: Hàm releaseETH trong **Hold** có thể bị hủy nếu trạng thái permittedAddresses chưa được định nghĩa bởi hàm addAddress => từ phân tích ICDF ta sẽ tìm kiếm các hàm có thể định nghĩa hoặc thay đổi biến trạng thái cần thiết ngăn cho giao dịch bị hủy

4.3

CrossFuzz thực hiện kiểm thử fuzz đa hợp đồng bằng cách tạo giao dịch gọi hàm từ hợp đồng cần kiểm thử và các hợp đồng phụ thuộc, rồi gửi chúng đến Máy ảo Ethereum (EVM) để phát hiện lỗ hổng bảo mật. Quá trình gồm bốn bước: triển khai hợp đồng, tạo trường hợp thử nghiệm, thực thi thử nghiệm, và đột biến thử nghiệm

- Đầu tiên, các hợp đồng phụ thuộc được triển khai trên Máy ảo Ethereum (EVM), và CrossFuzz lưu lại các địa chỉ của chúng để sử dụng trong quá trình thử nghiệm.. Tiếp theo, nó triển khai hợp đồng cần kiểm thử với các tham số khởi tạo được tạo ở phần 4.1

- Tiếp theo thì dựa trên Application Binary Interface (ABI) của hợp đồng thì ABI cung cấp thông tin về tên hàm, kiểu tham số và kiểu giá trị trả về. Khi đó CrossFuzz sử dụng thông tin này để tạo ra các giao dịch thử nghiệm cơ bản (một lần gọi hàm cho mỗi hàm trong hợp đồng hoặc các hợp đồng phụ thuộc)

Ví dụ : Giao dịch 1: Gọi hàm addAddress của PermissionManager, Giao dịch 2: Gọi hàm releaseETH của Hold

- Sau đó, CrossFuzz mã hóa các chuỗi giao dịch thành dữ liệu hệ thập lục phân rồi gửi đến EVM để thực thi. Trong quá trình thực thi, nó giám sát các thay đổi trong ngăn xếp, lượng gas tiêu thụ, môi trường khối, và các thông tin khác để phát hiện lỗ hổng bảo mật

+ Nếu có các điều kiện không đạt gây lỗi giao dịch thì lệnh REVERT của EVM sẽ được kích hoạt để hủy thay đổi và kết thúc giao dịch. Đối với các giao dịch bị kết thúc bởi lệnh REVERT được coi là các giao dịch ngoại lệ làm giảm khả năng bao phủ mã và cản trở việc thực thi các giao dịch tiếp theo cho nên thông qua ICDF ta sẽ tái sử dụng các giao dịch này. Cụ thể:

+ Thông qua define\_map thì CrossFuzz tìm ra các hàm có khả năng định nghĩa lại biến trạng thái được sử dụng trong giao dịch bị lỗi

+ Tính điểm ưu tiên cho các hàm tiềm năng



+ trong đó Sdef là số biến trạng thái được hàm f định nghĩa ma chưa được định nghĩa trước giao dịch lỗi, Sprovide là số biến trạng thái mà giao dịch lỗi cần sử dụng, Suse là số biến trạng thái mà hàm sử dụng nhưng không liên quân đến lỗi

+ Đưa lại hàm ưu tiên vào chuỗi giao dịch => giúp giao dịch lỗi vượt qua revert

Giả sử giao dịch releaseETH trong hợp đồng Hold bị lỗi **REVERT** do permittedAddresses chưa được định nghĩa:

1. **define\_map:**
   * addAddress định nghĩa biến permittedAddresses.
2. **Tính điểm ưu tiên:**
   * SDef(addAddress, releaseETH): 1 (định nghĩa permittedAddresses).
   * SProvide(addAddress, releaseETH): 1 (cung cấp permittedAddresses).
   * SUse(addAddress, releaseETH): 0 (không sử dụng biến không cần thiết).
   * **Tổng điểm:** S(addAddress,releaseETH)=1+1−0=2S(addAddress, releaseETH) = 1 + 1 - 0 = 2S(addAddress,releaseETH)=1+1−0=2.
3. **Kết quả:**
   * addAddress được chọn và thêm vào chuỗi giao dịch trước khi gọi releaseETH