HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

BÀI GIẢNG MÔN HỌC

An toàn hệ điều hành

Đánh giá an toàn

Giảng viên: TS. Phạm Hoàng Duy

Điện thoại/E-mail: phamhduy@gmail.com

Bộ môn: An toàn thông tin - Khoa CNTT1

Các kỹ thuật kiểm chứng mã chương trình

Giới thiệu

- Tất cả các dự án phần mềm tối thiểu đều hướng tới ít nhất một kết quả là đoạn mã. Ở mức đoạn mã, mục tiêu trọng tâm là các lỗi trong việc triển khai, nhất là những lỗi mà công cụ quét mã nguồn với những lỗ hổng phổ biến có thể phát hiện ra được.
  - Các lỗi triển khai vừa nhiều vừa phổ biến và gồm cả những lỗi nổi tiếng như tràn bộ đệm.
- Quá trình đánh giá mã nguồn cả thủ công hay tự động đều nhằm mục đích xác định các lỗi liên quan đến an ninh hay an toàn trước khi phần mềm được xuất xưởng.

Giới thiệu

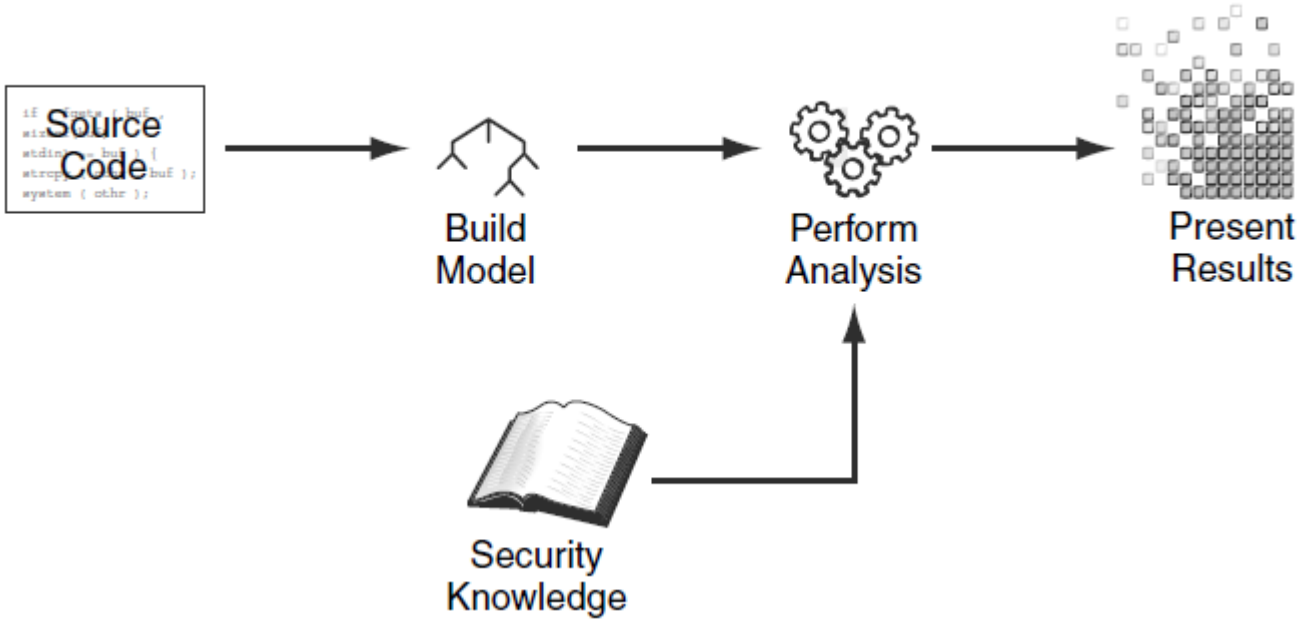
- Tất nhiên, không có phương thuốc trị bách bệnh. Việc đánh giá mã nguồn là cần thiết nhưng chưa đủ để đạt được phần mềm an ninh.
  - Các lỗi an ninh là các vấn đề hiển hiện song các lỗ hổng thiết kế còn là vấn đề nghiêm trọng hơn và hầu như không thể phát hiện nhờ việc đánh giá mã nguồn. Các tiếp cận đầy đủ để đạt được phần mềm an toàn là kết hợp hài hòa giữa đánh giá mã nguồn và phân tích thiết kế.
  - Việc đánh giá mã nguồn hiển nhiên cần những tri thức về lập trình. Việc hiểu biết các cớ chế an toàn hay an toàn mạng không có ích nhiều với việc đánh giá mã nguồn.

Phân tích tĩnh

Giới thiệu

- Phân tích tĩnh đề cập đến các kỹ thuật đánh giá mã nguồn nhằm cảnh báo các lỗ hổng an toàn tiềm tàng mà không thực thi chúng.
  - Một cách lý tưởng các công cụ tự động có thể tìm kiếm các lỗi an ninh với mức độ đảm bảo nhất định về các lỗi này song việc này vượt quá khả năng của rất nhiều công cụ hiện thời.
  - Các kỹ thuật kiểm tra phần mềm (testing) thông thường nhằm kiểm tra các hành vi (chức năng) với người dùng thông thường trong điều kiện thông thường nên rất khó để phát hiện ra các lỗi liên quan đến vấn đề an ninh và an toàn.

Mô hình phân tích



Mô hình phân tích

- Việc đầu tiên công cụ phân tích tĩnh cần làm là chuyển mã chương trình thành *mô hình chương trình (program model)*
  - Mô hình chương trình thực chất là cấu trúc dữ liệu biểu diễn đoạn mã.
- Các kỹ thuật phân tích bao gồm
  - Phân tích từ vựng (lexical analysis)
  - Phân tích câu (parsing)
  - Cú pháp khái quát (astract syntax)
  - Phân tích ngữ nghĩa (semantic analysis)
  - Phân tích luồng điều khiển (control flow)
  - Phân tích luồng dữ liệu (data flow)
  - Phân tích lan truyền lỗi (Taint propagation)
  - Phân tích con trỏ ảo(pointer aliasing)

Phân tích từ vựng

- Chuyển đoạn mã thành chuỗi các thẻ (token) nhằm loại bỏ những thành phần không quan trọng trong đoạn mã chương trình
- Các thẻ có thể chứa định danh (biến, tên hàm, ...) và vị tri của chúng trong đoạn mã

```
if (ret) // probably true
  mat[x][y] = END_VAL;

IF LPAREN ID(ret) RPAREN ID(mat) LBRACKET ID(x) RBRACKET LBRACKET
ID(y) RBRACKET EQUAL ID(END_VAL) SEMI
```

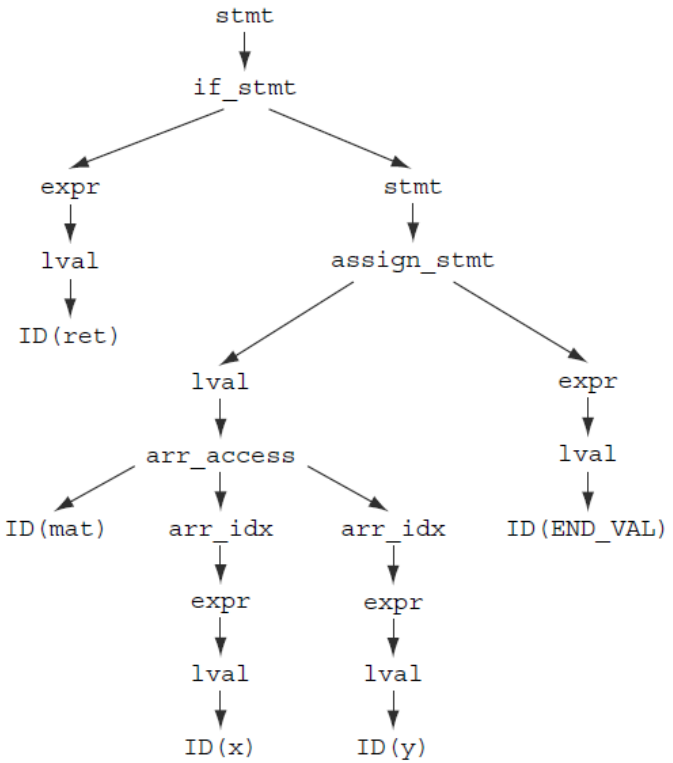
Phân tích câu

- Bộ phân tích câu sử dụng ngữ pháp phi ngữ cảnh (Context-free Grammar-CFG) để đối sánh các chuỗi từ hay thẻ. Bộ ngữ pháp sử dụng các luật sinh để diễn tả các ký hiệu của ngôn ngữ (lập trình)

```
stmt := if_stmt | assign_stmt
if_stmt := IF LPAREN expr RPAREN stmt
expr := lval
assign_stmt := lval EQUAL expr SEMI
lval = ID | arr_access
arr_access := ID arr_index+
arr_idx := LBRACKET expr RBRACKET
```

Phân tích câu

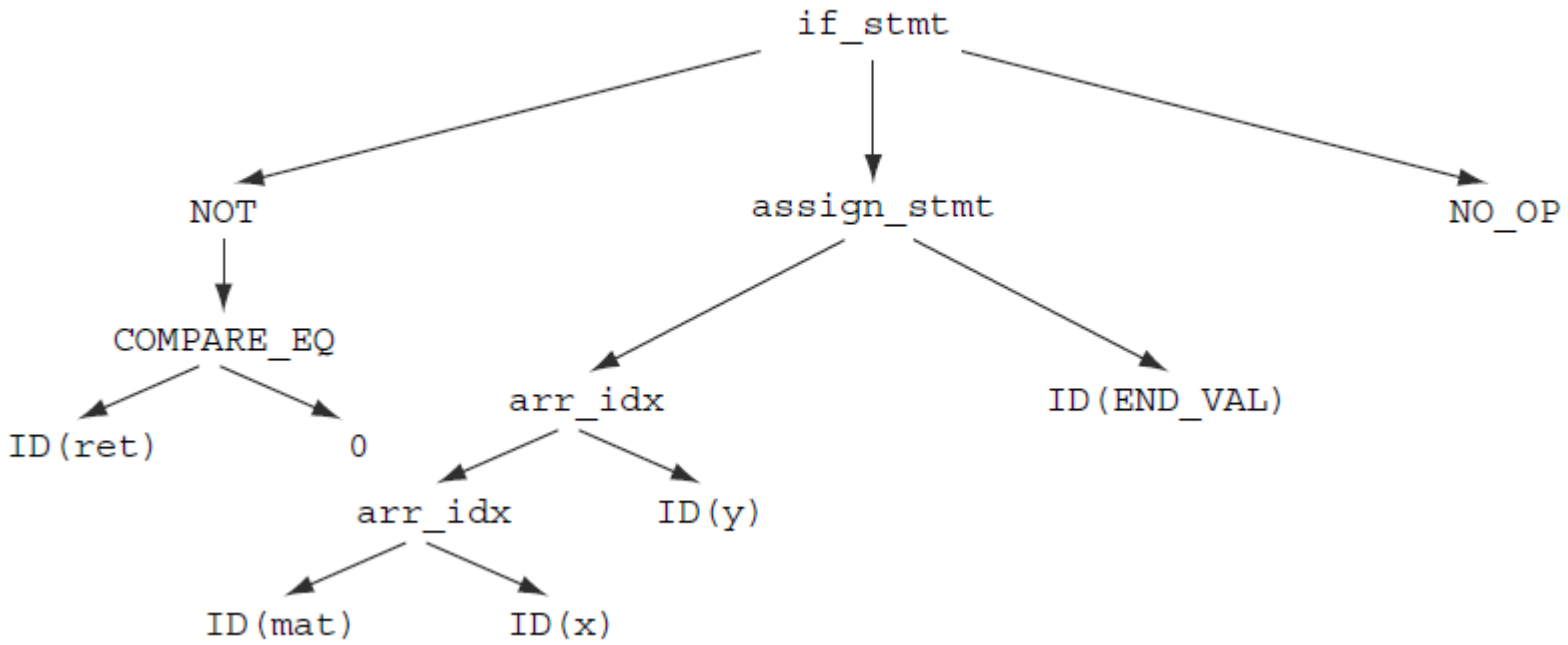
- Bộ phân tích câu thực hiện viêc suy diễn bằng cách đối sánh chuỗi các ký hiệu (thẻ) với các luật sinh để tạo ra các cây phân tích



Cú pháp khái quát

- Việc phân tích phức tạp có thể không phù hợp trên cây phân tích do mục tiêu của cây phân tích chỉ nhằm vào việc tách từ
- Cú pháp khái quát cung cấp cấu trúc dữ liệu phù hợp với việc phân tích tiếp theo bằng cách loại bỏ các ký hiêu (thẻ) không phù hợp. Các ký hiệu (thẻ) trong cú pháp khái quát có thể ít hơn so với ngôn ngữ lập trình ban đầu.

Cú pháp khái quát

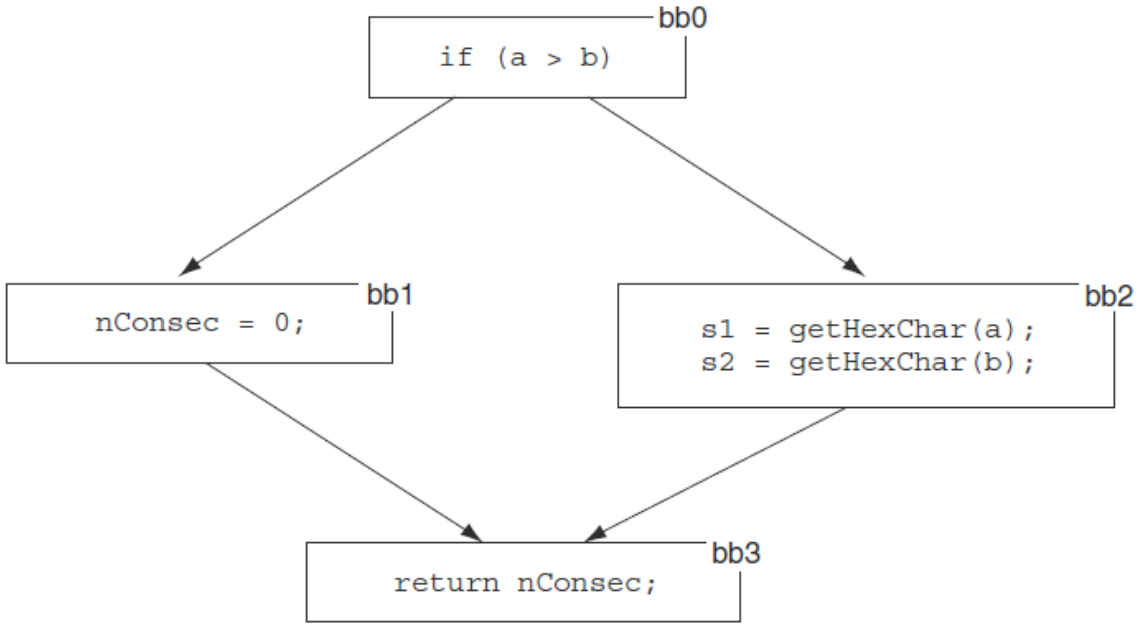


Phân tích ngữ nghĩa

- Việc phân tích ngữ nghĩa có thể được bắt đầu bằng việc phân rã các ký hiệu (tên biến hàm) và kiểm tra kiểu dữ liệu
  - Việc phân tích này cho phép lập cấu trúc đoạn mã thông qua việc kiểm tra các lớp của đối tượng được sử dụng, đặc biệt hữu ích với lập trình hướng đối tượng.

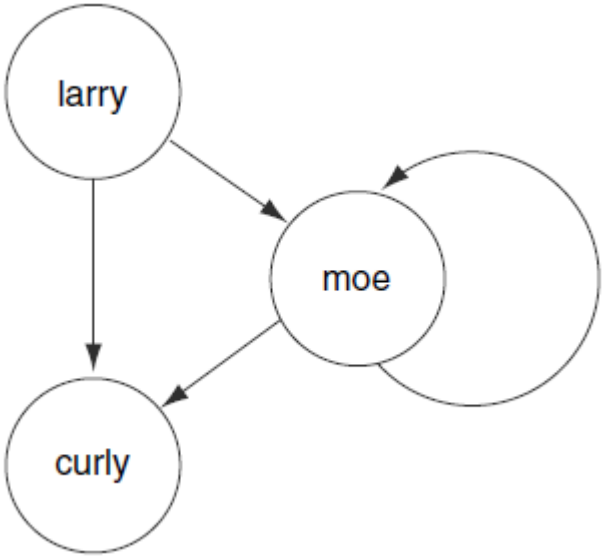
Phân tích luồng điều khiển

- Cho phép theo dõi các tình huống thực thi khác nhau của đoạn mã.
- Thường được thực hiện bằng cách xây dựng đồ thị luồng điều khiển



Phân tích luồng điều khiển

- Đồ thị gọi hàm biểu diễn luồng điều khiển giữa các hàm hay phương thức



Phân tích luồng dữ liệu

- Cho phép kiểm chứng cách dữ liệu di chuyển trong đoạn mã.
- Việc phân tích này thường kết hợp với luồng điều khiển để xác định vị trí bắt đầu và kết thúc của dữ liệu
- Việc phân tích luồng có thể phát hiện những tình huống như sử dụng mật khẩu hay khóa cố định trong đoạn mã

Phân tích lan truyền lỗi

- Sử dụng để tìm hiểu các giá trị bên trong đoạn mã mà người tấn công có khả năng kiểm soát bằng cách sử dụng luồng dữ liệu
  - Việc này cần thông tin về việc biến chứa lỗi xuất hiện ở đâu trong chương trình và cách thức di chuyển trong chương trình

Phân tích con trỏ ảo

- Mục đích của việc phân tích này là để hiểu cách thức các con trỏ có thể tham chiếu đến cùng vị trí nhớ. Việc phân tích này rất quan trọng với việc phân tích lan truyền lỗi

```
p1 = p2;
*p1 = getUserInput();
processInput(*p2);
```

Phân tích động

Giới thiệu

- Phân tích động phần mềm được thực hiện bằng cách chạy các đoạn mã trên bộ xử lý vật lý hay ảo
  - Các đoạn mã ngày càng sử dụng thư viện liên kết động hay theo yêu cầu, việc phân tích tĩnh không thể hiện đầy đủ các khía cạnh của đoạn mã
- Phân tích động nên được thực hiện sau khi hoàn thành việc phân tích tĩnh do việc phân tích động có thể làm tổn hại đến hệ thống
- Để phân tích và đánh giá tính an toàn của hệ thống, có thể cần thực hiện thêm:
  - Quét lỗ hổng
  - Kiểm thử xâm nhập

Sửa lỗi

- Việc phân tích động có thể được thực hiện thông qua phần mềm sửa lỗi debugger
  - Sửa lỗi có thể hoạt động ở mức cao, mức mã nguồn hay mức thấp hơn như mã máy
- Công cụ sửa lỗi cung cấp các cơ chế bẫy như chạy từng bước, chạy qua hay lấy thông tin về chương trình tại thời điểm chạy