HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

BÀI GIẢNG MÔN HỌC

An toàn hệ điều hành

Kiểm chứng mã chương trình

Giảng viên: TS. Phạm Hoàng Duy

Điện thoại/E-mail: phamhduy@gmail.com

Bộ môn: An toàn thông tin - Khoa CNTT1

- No Read-up A1:No-read-up, which means that subjects at the Unclassified-level are not able to read objects at the Classified-level*/
  - fact {no ((Classified.~sec).~R & UNClassified.~level)}
- Meta Policy-A-2: No-write-down, which prevents subjects at the Classified level to write objects at the Unclassified level.*/
  - fact {no ((UNClassified.~sec).~W & Classified.~level)}

Các điểm cần chú ý

- Phân tích tĩnh được sử dụng nhiều hơn chúng ta hình dung một phần do có nhiều công cụ và được sử dụng với nhiều mục đích khác nhau. Tiêu biểu
  - Kiểm tra kiểu
  - Kiểm tra cách lập trình
  - Hiểu chương trình
  - Kiểm chứng chương trình
  - Kiểm tra các thuộc tính
  - Tìm lỗi
  - Đánh giá an ninh

Kiểm tra kiểu

- Đây là hình thức được sử dụng nhiều nhất của phân tích tĩnh và quen thuộc với tất cả các lập trình viên

```
$ javac bar.java
bar.java:12: possible loss of precision
found   : int
required: short
  short r = i;
            ^
1 error
```

Kiểm tra cách lập trình

- Thường liên quan đến việc tuân thủ các yêu cầu về đặt tên, cấu trúc lập trình hay tương tự. Các vấn đề về cách lập trình thường ảnh hưởng đến việc hiểu chương trình cũng như việc bảo trì.

```
1 typedef enum { red, green, blue } Color;
2
3 char* getColorString(Color c) {
4  char* ret = NULL;
5  switch (c) {
6    case red:
7      printf("red");
8  }
9  return ret;
10 }
```

```
enum.c:5: warning: enumeration value 'green' not handled in switch
enum.c:5: warning: enumeration value 'blue' not handled in switch
```

Hiểu chương trình

- Các công cụ dạng này giúp cho người dùng nắm bắt được ý nghĩa của chương trình trong kho phần mềm lớn.
  - Một số công cụ cho phép phân tích các chương trình đoạn mã ở mức cao về tổ chức và cấu trúc của các hàm, các lớp và vị trí thực hiện lời gọi các hàm này tương tự như việc tái tạo hồ sơ thiết kế của các chương trình.

Kiểm chứng chương trình và thuộc tính

- Đánh giá các đoạn mã thực thi đầy đủ các đặc tả của chương trình. Nếu các đặc tả về chương trình bao trùm toàn bộ các chức năng của chương trình, các công cụ kiểm chứng có thể thực hiện việc kiểm chứng tương đương để chắc chắn đoạn mã và chương trình tương ứng chính xác với nhau.
- Thông thường, các đặc tả (chính tắc) chỉ áp dụng cho một số chức năng thiết yếu của chương trình (hệ thống) nên chỉ áp dụng việc kiểm chứng một phần. Đôi khi còn được gọi là kiểm tra thuộc tính. Đây là các điều kiện mà chương trình phải tuân theo.

Kiểm chứng chương trình và thuộc tính

```
1  inBuf = (char*) malloc(bufSz);
2  if (inBuf == NULL)
3    return -1;
4  outBuf = (char*) malloc(bufSz);
5  if (outBuf == NULL)
6    return -1;
```

```
Violation of property "allocated memory should always be freed":
  line 2: inBuf == NULL
  line 3: function returns (-1) without freeing inBuf
```

Tìm lỗi

- Mục tiêu là chỉ ra tình huống mà chương trình có thể hoạt động không như người lập trình dự định. Công cụ tìm lỗi lý tưởng là đủ (chứng minh đủ) cần cung cấp phản ví dụ. Điều này cho biết kết quả có thể khi xảy ra lỗi.

Đánh giá an ninh

- Các công cụ thời ký đầu tập trung vào việc quét các hàm hay bị lạm dụng như strcpy và cần tiến hành việc đánh giá an ninh thủ công. Điều này đôi khi là cho người dùng coi các lỗi phát hiện được như là các lỗi lập trình hơn là các vấn đề an ninh cần được chú ý.
- Các công cụ mới thường kết hợp việc tìm lỗi và kiểm tra thuộc tính. Việc kiểm tra tràn bộ đệm có thể được diễn giải như là yêu cầu lập trình sao cho chương trình không truy nhập ngoài không gian nhớ được cấp phát.

Đánh giá an ninh

- Đoạn mã dưới đây chứa câu lệnh không an toàn. Vì sao?

```
int main(int argc, char* argv[]) {
  char buf1[1024];
  char buf2[1024];
  char* shortString = "a short string";
  strcpy(buf1, shortString); /* safe use of strcpy */
  strcpy(buf2, argv[0]);     /* dangerous use of strcpy */
  ...
```

Type of Tool/Vendors	Web Site
Style Checking	
PMD	http://pmd.sourceforge.net
Parasoft	http://www.parasoft.com
Program Understanding	
Fujaba	http://wwwcs.uni-paderborn.de/cs/fujaba/
CAST	http://www.castsoftware.com
Program Verification	
Praxis High Integrity Systems	http://www.praxis-his.com
Escher Technologies	http://www.eschertech.com

Type of Tool/Vendors	Web Site
Bug Finding	
FindBugs	http://www.findbugs.org
Coverity	http://www.coverity.com
Visual Studio \analyze	http://msdn.microsoft.com/vstudio
Klocwork	http://www.klocwork.com
Security Review	
Fortify Software	http://www.fortify.com
Ounce Labs	http://www.ouncelabs.com