LẬP TRÌNH AN TOÀN
TRONG PHÁT TRIỂN ỨNG DỤNG WEB

Secure Programming
Vũ Minh Tuấn
vu.tuan@jvb-corp.com

MỤC ĐÍCH

- ✓Giới thiệu lập trình an toàn
  - ✓Bảo mật là gì, tại sao cần lập trình an toàn
  - ✓Phương thức tấn công, các lỗi thường gặp.
  - ✓Cách phòng chống, bảo mật.
- ✓Đưa ra các yêu cầu về lập trình an toàn
- ✓Ví dụ thực hành, bài tập.

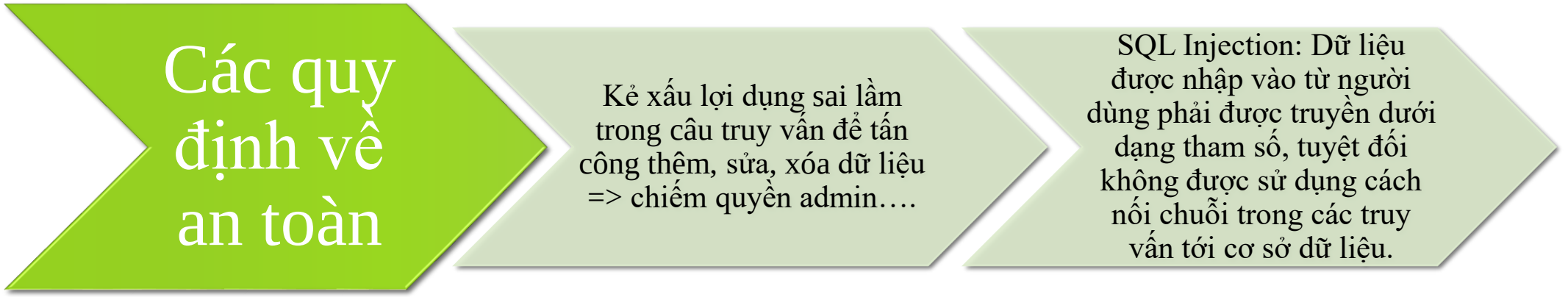
GIỚI THIỆU VỀ BẢO MẬT

- Bảo mật thông tin (information security): Bảo vệ các thông tin ở dạng số hóa: Thông tin cung cấp giá trị cho con người và cho tổ chức.
- Tại sao cần:
  - Phòng ngừa đánh cắp dữ liệu
  - Tránh các hậu quả liên quan tới pháp luật
  - Duy trì quá trình hoạt động của ứng dụng

GIỚI THIỆU VỀ BẢO MẬT

- Các phương thức tấn công, lỗi thường gặp:
  - SQL Injection *
  - XSS *
  - CSRF *
  - Kiểm soát các thao tác với file *
  - Mã hóa dữ liệu nhạy cảm *
  - Kiểm tra quyền truy cập của người dùng *
  - User enumeration *
  - Session fixation
  - Session Hijacking
  - HTTP Only cookie
  - Chuyển hướng và chuyển tiếp thiếu thẩm tra
  - Để lộ dữ liệu hệ thống
  - Lộ thông tin do kiểm tra ngoại lệ không tốt
  - File inclusion
  - Command injection

SQL Injection *



SQL Injection *

Example

```
txtUserId = getRequestString("UserId");
txtSQL = "SELECT * FROM Users WHERE UserId = " + txtUserId;
```

SQL Injection Based on 1=1 is Always True

Look at the example above again. The original purpose of the code was to create an SQL statement to select a user, with a given user id.

If there is nothing to prevent a user from entering "wrong" input, the user can enter some "smart" input like this:

UserId: 105 OR 1=1

Then, the SQL statement will look like this:

```
SELECT * FROM Users WHERE UserId = 105 OR 1=1;
```

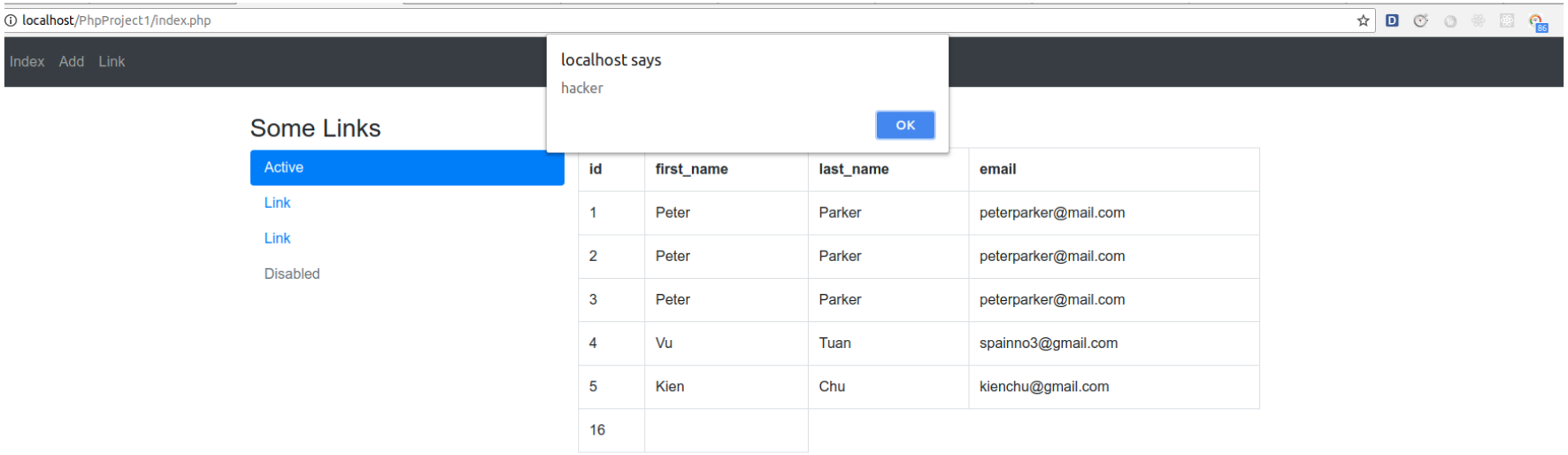
XSS *

Các quy định về an toàn

XSS là kỹ thuật tấn công bằng cách chèn vào dữ liệu đầu ra các web động, thẻ html, các đoạn mã nguy hiểm thường là Client-site Script

XSS: Encode dưới dạng html các ký tự đặc biệt do client gửi đến bao gồm: <,>,&,’,”,/ trong các trường hợp gửi lên hoặc dữ liệu lấy ra.

XSS *


localhost/PhpProject1/index.php
Index Add Link
localhost says
hacker
OK
Some Links
Active
Link
Link
Disabled
id first_name last_name email
1 Peter Parker peterparker@mail.com
2 Peter Parker peterparker@mail.com
3 Peter Parker peterparker@mail.com
4 Vu Tuan spainno3@gmail.com
5 Kien Chu kienchu@gmail.com
16

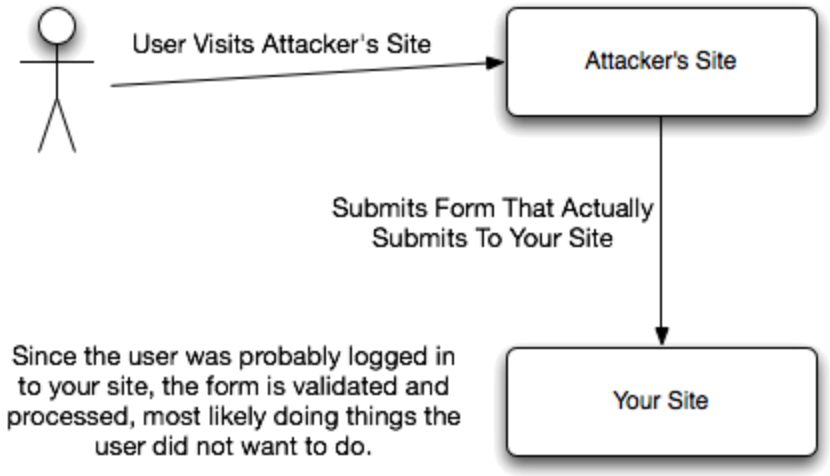
CSRF *

Các quy định về an toàn

Là phương pháp mượn quyền của người khác để thực hiện các hành động không được phép.

CSRF: Đối với các yêu cầu làm ảnh hướng đến dữ liệu thì phải sử dụng thêm token. Server sẽ phải kiểm tra token gửi lên từ client

CSRF *


User Visits Attacker's Site
Attacker's Site
Submits Form That Actually
Submits To Your Site
Since the user was probably logged in
to your site, the form is validated and
processed, most likely doing things the
user did not want to do.
Your Site

Kiểm soát thao tác với file upload *

Các quy định về an toàn

Nếu ứng dụng không kiểm soát thì dẫn tới việc download,upload những file không hợp lệ.

File upload: Kiểm tra phần mở rộng file, lọc bỏ các ký tự /,\,null trong tên file, tên file cần được sinh ngẫu nhiên.

File name: ./../..config.ini

Mã hóa dữ liệu nhạy cảm *

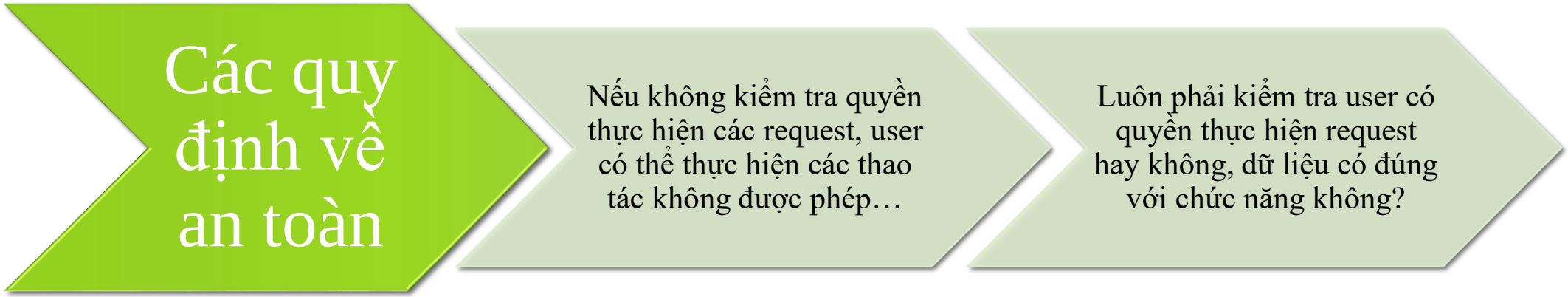
Các quy định về an toàn

Nếu hệ thống bị tấn công, kẻ gian sẽ lấy được những thông tin chưa được mã hóa hoặc mã hóa không an toàn như password, thẻ ngân hang…

Mã hóa dữ liệu: mã hóa dữ liệu nhạy cảm như password bằng các hàm mã hóa 1 chiều với salt

Password, hidden field …

Kiểm tra quyền truy cập *

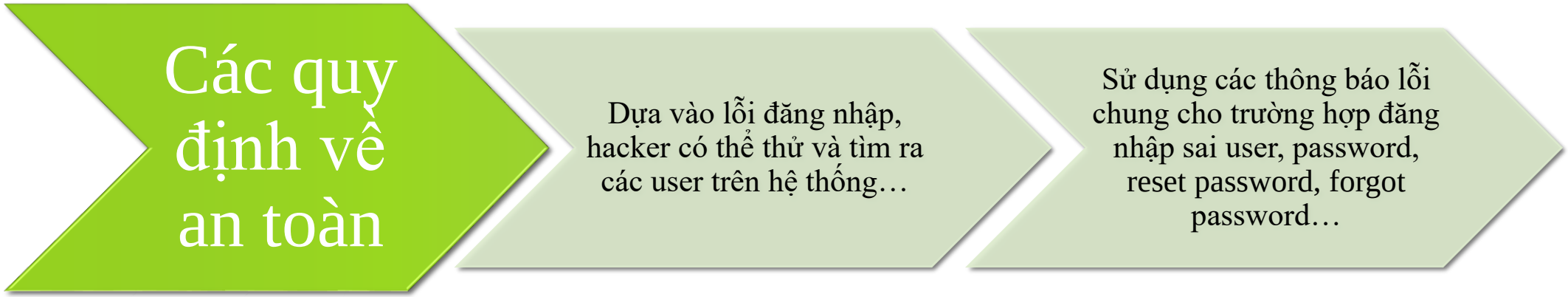


Kiểm tra quyền truy cập *

```php
public function deleteFileOfuser() {

    $idFile = (int) $data['id'];

    doDeleteFile($idFile);

    return SUCCESS;

}
```

```php
public function deleteFileOfuser() {
    $idFile = (int) $data['id'];

    if(checkPermission ($userId)) {
        doDeleteFile($idFile);
        return SUCCESS;
    }
    return ERROR;
}
```

User enumeration *



User enumeration

```php
private function checkLogin(string $email, string $password) {

    if (!empty($user) && !equals($user)) {
        return 'User khong ton tai';
    }

    if (!empty($password) && !equals($password)) {
        return 'Sai password';
    }

    return 'Success';
}
```

Session fixation & Hijacking

Các quy định về an toàn

Là cách tấn công dựa vào việc gửi 1 sessionId giả mạo, hoặc mạo danh bằng việc giải mã sessionId được lưu ở cookie hoặc qua biến ẩn …

Luôn khởi tạo session mới khi người dung đăng nhập, xóa bỏ session khi người dung logout, thiết lập thời gian cho session …

Session_regenerate_id();

Sử dụng cookie an toàn

Redirect thiếu thẩm tra

Các quy định về an toàn

Hacker có thể lừa người dùng redirect đến URL nhiễm mã độc, lừa nạn nhân để lấy user, password…

Hạn chế redirect và chuyển hướng đến các URL khác. Hạn chế sử dụng tham số được redirect. Kiểm tra tham số trước khi redirect.

https://www.google.com/?redirect.php?url=jvb-corp.com

Để lộ dữ liệu hệ thống

Các quy định về an toàn

Để tồn tại và cho phép truy xuất các file trên hệ thống, ví dụ khi làm việc với file excel, csv nhưng không xóa và vẫn có thể xem các file này

Các dữ liệu phải để bên ngoài thư mục cài đặt web server, việc download phải qua action và có xác thực và tham số mã hóa.

https://www.google.com/share/data/list_all_admins.xls

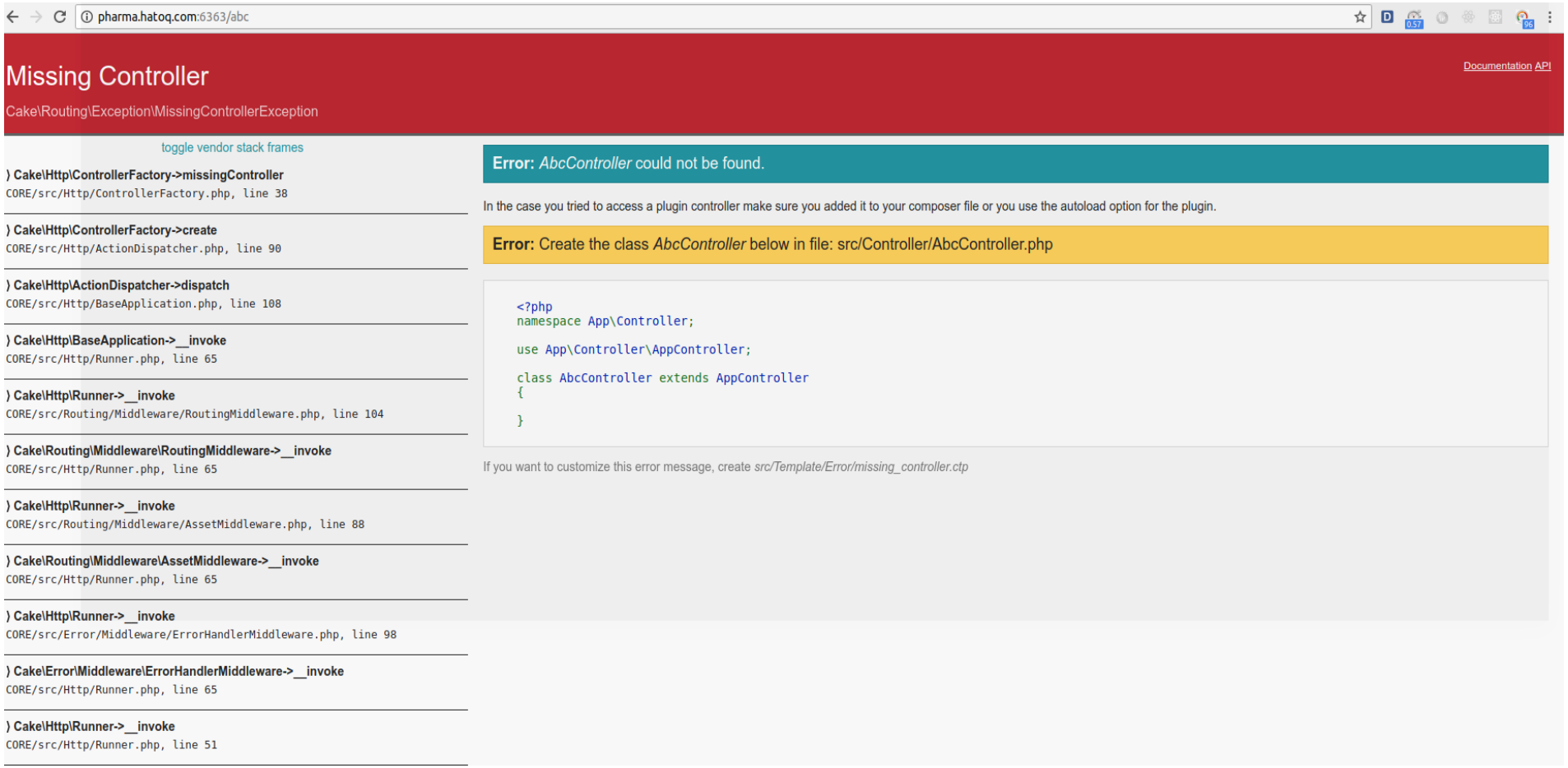
Kiểm tra ngoại lệ

Các quy định về an toàn

Việc hiển thị quá chi tiết và nhiều lỗi khi xử lý (bug), giúp hacker có thể đoán biết được hệ thống cũng như thông tin tiếp cận lỗ hổng…

Các ngoại lệ phải được lưu vào log để xử lý sau, không được hiển thị chi tiết lỗi phía người dùng…

Kiểm tra ngoại lệ



File inclusion

Các quy định về an toàn

Do không kiểm soát tốt việc include các file, class thì hacker có thể chuyển hướng gọi tới các file chứa mã độc

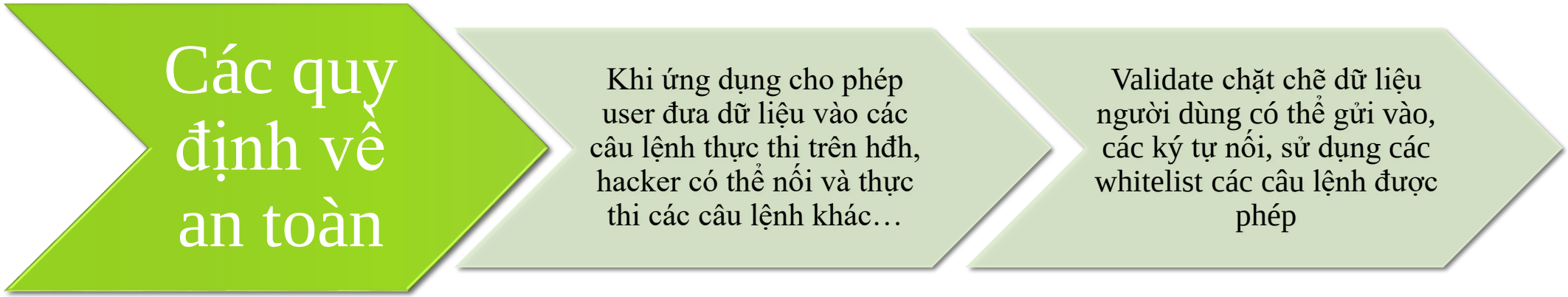
Phân quyền thư mục hợp lý. Không cho phép include remote nếu không cần thiết, thiết lập php.ini : allow_url_fopen=off

Ví dụ sử dụng file làm nội dung: index.php?page=contact.html

File inclusion

```php
$whiteList = array('index.html, download.html, info.html');
if(in_array($_GET['page'], $whiteList)) {
    include($_GET['page']);
} else {
    die('Attack attempt');
}
```

Command injection



Command injection

```php
//exc trực tiếp với param
system("nslookup $param");

//Kiểm tra trước khi thực hiện
system("nslookup ". escapeshellarg($param));

//OR
system("nslookup ". preg_replace("/[^a-z0-9\-\.]/i", "", $param));
```

Ứng dụng trong các framework

CakePHP

- SecurityComponent (HTTP methods, CSRF, Form tampering prevention, SSL, hash, encrypt …
- Function h
- ….
- => Not sure 100%

Laravel

- Hash, CSRF Protection, Encrypting
- Blade view {{ }}
- ….
- => Not sure 100%

Tóm lại một số cách phòng chống

- Sử dụng https, ssl, tls
- Vô hiệu hóa các hàm quan trọng
- Sao lưu (backups) thường xuyên
- Mã hóa file cấu hình
- Sử dụng xác thực (nếu cần)
- Phân quyền hợp lý
- Kiểm soát input
- Hạn chế truy cập
- Tắt error_reporting
- Tránh đặt mật khẩu dễ đoán
- Cập nhật tin tức về lỗ hổng, bản vá
- Plugin chính chủ
- Quét mã độc
- Firewall
- Session, cookie an toàn
- **Đừng bao giờ tin tưởng người dùng**
- .....
- Hỏi ý kiến chuyên gia

Hỏi đáp

Bài tập thực hành

Bài tập thực hành

Bài tập thực hành: Viết đoạn mã mô tả lỗi CSRF và 1 đoạn mã sau khi đã fix lỗi CSRF bằng việc sử dụng TOKEN

TÀI LIỆU THAM KHẢO

- ✓ https://php.earth/docs/security/intro
- ✓ http://www.phptherightway.com/
- ✓ https://www.owasp.org/index.php/PHP_Security_Cheat_Sheet
- ✓ https://phpsecurity.readthedocs.io/en/latest/
- ✓ https://www.google.com/