OWASP
Open Web Application
Security Project

安全代码审查

S-SDLC与安全代码审查

主讲人：咸士强

个人介绍

咸士强

- OWASP吉林区域负责人
- 嘉诚信息安全实验室负责人
- 守天众测平台负责人
- 欧米伽安全团队创始人

目录/CONTEN

OWASP
Open Web Application Security Project

OWASP
Open Web Application Security Project

S-SDLC

1.简介

2.描述

3.目标

S-SDLC

简介

S-SDLC是安全软件开发生命周期，是一套完整的，面向Web和APP开发厂商的安全工程方法。帮助软件企业降低安全问些，提升软件安全质量；

描述

S-SDLC定义了安全软件开发的流程，以及各个阶段需要进行的安全活动，包括活动指南，工具、模板等。

S-SDLC

OWASP
Open Web Application Security Project

目标

1.制定面向Web和APP开发企业的安全开发流程；
2.制定及开发安全基础培训课程；
3.根据实践经验，输出各个安全活动的方法指导及模板；
4.制定WEB应用/移动应用安全设计指南；
5.制定安全编码（C/C++、JAVA、PHP，C#）；
6.将OWASP现有项目，如开发指南、测试指南融合到软件全开发体系中。

S-SDLC

安全代码审查存在于每个正式的软件安全开发周期（S-SDLC）

安全代码审查

1.考虑因素

2.度量指标

安全代码审查——考虑因素

风险

目的与背景

资源、时间和期限

编程语言

代码行数

安全代码审查——度量指标

OWASP
Open Web Application Security Project

- 代码行数
- 代码复杂度
- 功能点

OWASP
Open Web Application Security Project

常见漏洞与防护

1.SQL注入
2.XSS
3.代码注入
4.命令执行
5.文件包含
6.文件上传

常见漏洞与防护——SQL注入

OWASP
Open Web Application Security Project

通过把SQL命令插入到Web表单递交或输入域名或页面请求的查询字符串，最终达到欺骗服务器执行恶意的SQL命令。

常见漏洞与防护——SQL注入

OWASP Open Web Application Security Project

常规注入

```php
<?php
//including the Mysql connect parameters.
include("../sql-connections/sql-connect.php");
error_reporting(0);
// take the variables
if(isset($_GET['id']))
{
$id=$_GET['id'];
//logging the connection parameters to a file for analysi
$fp=fopen('result.txt','a');
fwrite($fp,'ID:'.$id."\n");
fclose($fp);

// connectivity


$sql="SELECT * FROM users WHERE id='$id' LIMIT 0,1";
$result=mysql_query($sql);
$row = mysql_fetch_array($result);
```

宽字节注入

```php
<?php

header("Content-type:text/html;charset=utf-8");

$id=@$_GET['id'];

if(isset($id)&&!empty($id)){

$id = addslashes($_GET['id']);//进行转义

$conn=mysql_connect('localhost','root','') or die ("false");

mysql_select_db("tests",$conn);

mysql_query("set names 'gbk' ",$conn);//指明数据库gbk处理，不安全的编码格式

$sql = "select * from test where id='".$id."'";

$result = mysql_query($sql, $conn);
```

常见漏洞与防护——SQL注入

OWASP
Open Web Application Security Project

结构预处理

PDO参数绑定代码

函数转义

intval
addslashes

参数规则验证

输入的合法性
输入的长度

禁用魔术引号

常见漏洞与防护——XSS

OWASP Open Web Application Security Project

存储型XSS

存储型XSS，持久化，代码是存储在服务器中的，如在个人信息或发表文章等地方，插入代码，如果没有过滤或过滤不严，那么这些代码将储存到服务器中，用户访问该页面的时候触发代码执行。这种XSS比较危险，容易造成蠕虫，盗窃cookie。

反射型XSS

非持久化，需要欺骗用户自己去点击链接才能触发XSS代码（服务器中没有这样的页面和内容），一般容易出现在搜索页面。

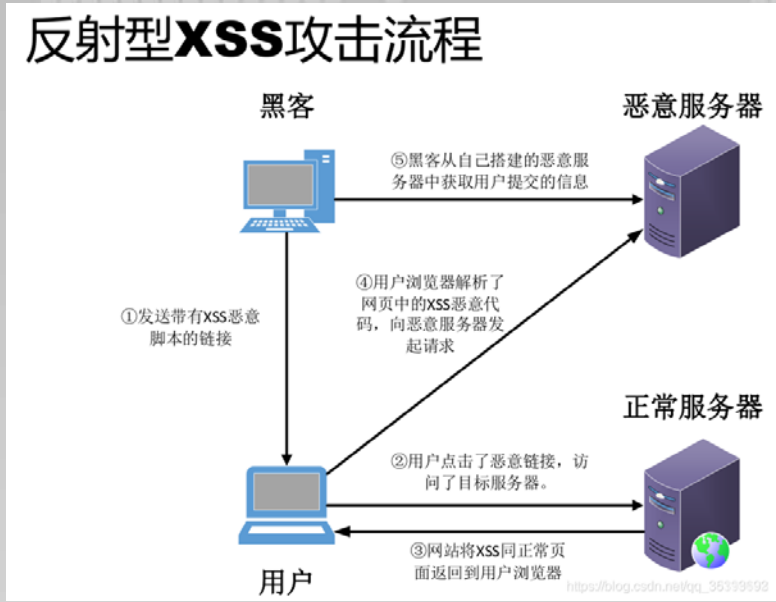
DOM型XSS

不经过后端，DOM-XSS漏洞是基于文档对象模型(Document Objeet Model,DOM)的一种漏洞，DOM-XSS是通过url传入参数去控制触发的，其实也属于反射型XSS。

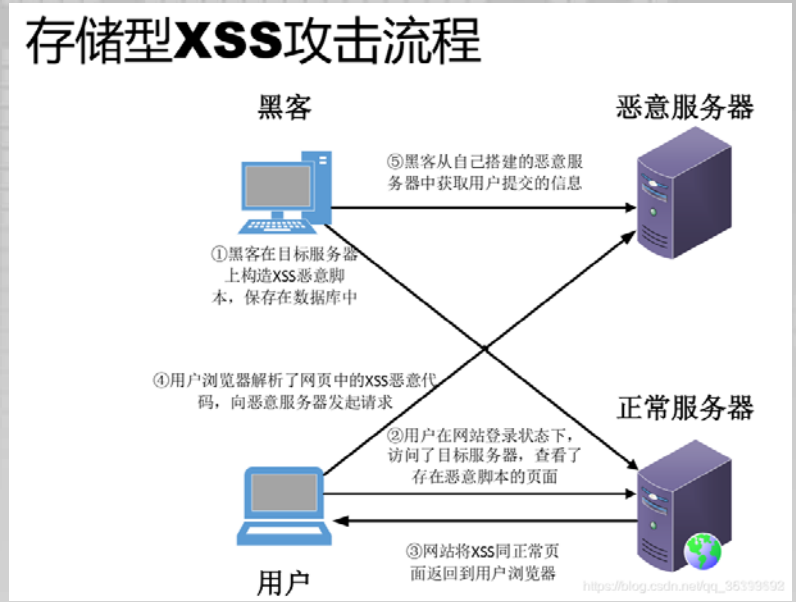
常见漏洞与防护——XSS

OWASP Open Web Application Security Project

反射型XSS



存储型XSS



常见漏洞与防护——XSS

OWASP Open Web Application Security Project

XSS的危害

常见漏洞与防护——XSS

OWASP Open Web Application Security Project

XSS防范

情况一
PHP直接输出html的，可以采用以下的方法进行过滤：

1.htmlspecialchars函数；
2.htmlentities函数；
3.HTMLPurifier.auto.php插件；
4.RemoveXss函数。

情况二
PHP输出到JS代码中，或者开发Json API的，则需要前端在JS中进行过滤：

1.尽量使用innerText(IE)和textContent(Firefox),也就是jQuery的text()来输出文本内容。
2.必须要用innerHTML等等函数，则需要做类似php的htmlspecialchars的过滤。

常见漏洞与防护——代码注入

OWASP Open Web Application Security Project

常规eval注入

eval()函数可以把字符串按照PHP代码来执行，换句话说，就是可以动态地执行PHP代码，使用eval函数需要注意的是：输入的字符串必须是合法的PHP代码，且必须以分号结尾。

```php
<?php
        $myvar = "varname";
        $x = $_GET['arg'];
        eval("\$myvar = $x;");
?>
```

攻击者可以通过如下Payload实施代码注入：/index.php?arg=1;phpinfo()

常见漏洞与防护——代码注入

eval单引号

开发者使用单引号包裹变量在一定程度上提升了安全性，但依旧存在问题。

```php
<?php
$data = $_GET['data'];
eval("\$ret = strtolower($data);");
echo $ret;
?>
```

攻击者可以通过如下Payload实施代码注入：/index.php?data=%27);phpinfo();//

常见漏洞与防护——代码注入

preg_replace /e 注入

开发者使用单引号包裹变量在一定程度上提升了安全性，但依旧存在问题。

```
<?
echo preg_replace("/test/e",$_GET["h"],"jutst test");
?>
```

攻击者可以通过如下Payload实施代码注入：/index.php?h=phpinfo()

常见漏洞与防护——代码注入

OWASP
Open Web Application Security Project

不把对象存储为字符串

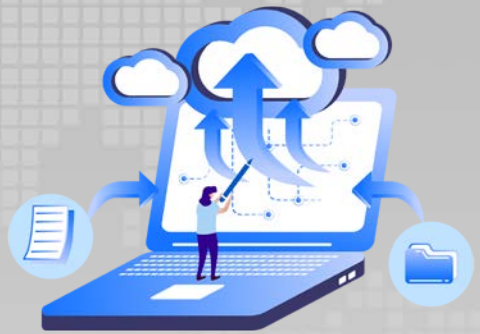
使用JSON来保持对象，而不是使用PHP序列化来保存。

慎用使用eval

一定使用单引号包裹可控代码，并在插入前进行addslashes。

弃用preg_replace /e 修饰符

换用preg_replace_callback来代替/e修饰符。



常见漏洞与防护——命令执行

常见漏洞与防护——命令执行

- 减少命令执行，能使用脚本解决的工作不要调用其他程序处理。
  尽量少使用执行命令的函数，并在disable_functions中禁用。

- 如果参数是用户所提供的，需要使用escapeshellarg函数进行过滤。

- 参数的值尽量使用引号包裹，并在拼接前调用addslashes。

常见漏洞与防护——文件包含

OWASP Open Web Application Security Project

常见导致文件包含漏洞函数

require():找不到被包含的文件，报错并且停止运行脚本。
include():找不到被包含的文件，只会报错，但会继续运行脚本。
require_once():与require()类似，区别在于重复调用同一脚本时，程序只调用一次。
include_once():与include()类似，区别在于重复调用同一函数时，程序只调用一次。

常见漏洞与防护——文件包含

- Include()等函数通过动态变量的方式引入需要包含的文件
- 攻击者能够控制该动态变量

常见漏洞与防护——文件包含

1.严格限制包含中的参数，取消那些不可控的参数。

2.开启open_basedir()函数，将其设置为指定目录，则只有该目录的文件允许被访问。

3.如果不需要文件包含， 则关闭allow_url_include()函数，防止远程文件包含，这是最安全的办法。

4.如果需要使用文件包含，则通过使用白名单的方法对要包含的文件进行限制，这样可以做到既使用了文件包含，又可以防止文件包含漏洞。

常见漏洞与防护——文件上传

OWASP
Open Web Application Security Project

- 系统开发阶段的防御
- 系统运行阶段的防御
- 系统维护阶段的防御

业务逻辑安全

1.验证码安全

2.业务流程安全

业务逻辑安全——验证码安全

OWASP
Open Web Application Security Project

1.验证码的字符范围要尽可能大

2.尽量让字符进行变形、扭曲，或使用干扰性强的图案

3.防暴力破解，要对生成的每个验证码设置一个有效期

4.防止生成的验证码返回到相应中

业务逻辑安全——验证码安全

- **短信验证码**
  - **问题**

    1.短信炸弹；
    2.经济损失；
    3.短信内容注入。
  - **解决方案**

    1.使用短信验证码时，在发送短信验证码时进行人机校验；
    2.限制单个手机号某个时间段内最多接收的短信数量；
    3.根据业务需求限制短信发送的时间段；
    4.防止用户直接或者间接的自定义短信内容。

业务逻辑安全——验证码安全

OWASP Open Web Application Security Project

- **语音验证码**
  - **认证方式**

    1.在认证页面进行播放；
    2.用户主动呼叫系统的预留电话获取验证码；
    3.由用户触发，系统通过拨打用户绑定电话接听验证码。

  - **解决方案**

    1.使用语音验证码时，进行人机校验；
    2.对验证码进行有效期的设置，在认证失败后将验证码进行失效处理，防止暴力破解；
    3.防止频繁请求，限制单个用户单个手机号某个时间段的认证次数。

业务逻辑安全——业务流程安全

OWASP Open Web Application Security Project

- **注册安全**
  - **问题**

    1.恶意注册；
    2.恶意遍历。

  - **解决方案**

    1.注册功能增加人机认证；
    2.记录IP的请求次数，控制单位时间内IP的请求次数，防止恶意遍历注册用户；
    3.不提供独立的用户名检测接口，防止被恶意利用。

业务逻辑安全——业务流程安全

OWASP
Open Web Application Security Project

- **登陆安全**
  - **问题**

    1.信息泄露；
    2.密码爆破。

  - **解决方案**

    1.手机号验证码登录；
    2.登陆错误提示建议使用“用户名或密码错误”；
    3.登录进行密码尝试频率限制；
    4.强制要求密码复杂度。

OWASP

Open Web Application Security Project

THANKS

主讲人：咸士强