PHP

1.SQL 注入

1.原理

常见的SQL注入漏洞主要是由于程序开发过程中不注意规范书写sql语句以及对特殊字符的不严格过滤，从而导致客户端可以通过全局变量POST或GET提交恶意代码。

2.修复方案：转义、报错、黑名单

1.利用黑名单

```
$filter =
"regexp|from|count|procedure|and|ascii|substr|substring|left|right|union|if|case|pow|ex
p|order|sleep|benchmark|into|load|outfile|dumpfile|load_file|join|show|select|update|se
t|concat|delete|alter|insert|create|union|or|drop|not|for|join|is|between|group_concat|
like|where|user|ascii|greatest|mid|substr|left|right|char|hex|ord|case|limit|conv|table
|mysql_history|flag|count|rpad|\&|\*|\.|-";
if((preg_match("/".$filter."/is",$username)== 1) ||
(preg_match("/".$filter."/is",$password)== 1)){ die(); }
```

2.利用转义

修改php.ini中的`magic_quotes_gpc`的配置，修改为ON(php>5.4)被删除了

利用`addslashes()`对特殊符号进行转义:单引号、双引号、反斜杠

但是需要注意的是 如果已经开启了`magic_quotes_gpc`，所有的特殊字符已经默认是用了`addslashes`这种情况下会导致双重转义。

可以利用`get_magic_quotes_gpc`进行检测，如果已经转义了，就不需要再次进行转义

3.报错

关闭报错信息，可以过滤掉直接回显的信息，可以防止大部分注入

2.反序列化

修复方案

1.利用PHP7新特性 为`unserialize`提供过滤

```
<?php
// 将所有的对象都转换为 __PHP_Incomplete_Class 对象
$data = unserialize($foo, ["allowed_classes" => false]);
// 将除 MyClass 和 MyClass2 之外的所有对象都转换为 __PHP_Incomplete_Class 对象
$data = unserialize($foo, ["allowed_classes" => ["MyClass", "MyClass2"]);
// 默认情况下所有的类都是可接受的，等同于省略第二个参数
$data = unserialize($foo, ["allowed_classes" => true]); ?>
```

2.限制session反序列化

`php_serialize`在5.5版本后添加了一种新的规则。

如果在5.4及之前版本，如果设置成`php_serialize`会报错

1.限制保存序列化和反序列化时使用的处理器

```
正确设置序列化及反序列化时使用的处理器
ini_set('session.serialize_handler', 'php_serialize');
ini_set('session.serialize_handler', 'php');
```

2.限制phar拓展php反序列化

```
$filter = "phar|zip|compress.bzip2|compress.zlib";

if(preg_match("/".$filter."/is",$name)== 1){
    die();
}
```

3.文件上传

修复方案

1.后端代码限制上传的文件类型和大小

```
if ((($_FILES["Up10defile"]["type"]=="image/gif")&&(substr($_FILES["Up10defile"]
["name"], strrpos($_FILES["Up10defile"]["name"], '.')+1))=='gif')&&($_FILES["file"]
["size"]<1024000){ } else{ die(); }
```

2.强制给上传的文件添加后缀名

3.利用白名单限制

```
$a = array('.png','.jpg','.gif');
if(in_array($a,$str)){
    ....
}
```

4.文件包含

修复方案

1.LFI(本地文件包含)

本地路径包含限制

```
$filename = $_GET['filename'];
$pattern = "\/|\.\.\/|\.\/|etc|var|php|jpg|jpeg|png|bmp|gif";
if(preg_match("/".$pattern."/is",$filename)== 1){
echo "die000000000000000000000000000000"; die();
}
include($filename);
```

2.RFI(远程路径包含限制)

远程路径包含限制

```
$filename = $_GET['filename'];
$pattern = "\/|\.\.\/|\.\/|etc|var|php|jpg|jpeg|png|bmp|gif";
if(preg_match("/".$pattern."/is",$filename)== 1){
  echo "die000000000000000000000000000000";
    die();
}
include($filename);
```

3.限制环境

```
allow_url_fopen = off   (是否允许打开远程文件)
allow_url_include = off (是否允许include/require远程文件)
```

4.php伪协议

协议过滤&路径访问限制

```
$filename = $_GET['filename'];
$pattern =
"\/|\.\.\/|\.\/|etc|var|php|jpg|jpeg|png|bmp|gif|file|http|ftp|php|zlib|data|glob|phar|
ssh2|rar|ogg|expect|zip|compress|filter|input";
if(preg_match("/".$pattern."/is",$filename)== 1){
echo "die0000000000000000000000000000000"; die();
} include($filename);
```

5.任意文件读取

修复方案

1.过滤常见的伪协议

```
$filename = $_GET['filename'];
$pattern =
"\/|\.\.\/|\.\/|etc|var|php|jpg|jpeg|png|bmp|gif|file|http|ftp|php|zlib|data|glob|phar|
ssh2|rar|ogg|expect|zip|compress|filter|input";
if(preg_match("/".$pattern."/is",$filename)== 1){
  echo "die0000000000000000000000000000";
    die();
}
include($filename);
```

2.利用`open_basedir`

限制文件访问范围

`open_basedir=/var/www/html`

6.命令执行

```
# 代码执行函数
eval()、assert()、call_user_func()、call_user_func_array()、array_map()等
# 正则处理
mixed preg_replace ( mixed $ pattern , mixed $ replacement , mixed $ subject [, int $
limit = -1 [, int &$ count ]] ) preg_replace() 参数/e修饰符问题
# 调用函数过滤不严
call_user_func()和array_map()
```

修复方案

1.php7特性

`preg_replace()`函数不在支持\e,利用`preg_replace_callback()`代替

2.对回调函数进行过滤

```
<?php
$a=$_GET['cc'];
$pattern =
"eval|assert|passthru|pcntl_exec|exec|system|escapeshellcmd|popen|chroot|scandir|chgrp|
chown|shell_exec|proc_open|proc_get_status|ob_start";
if(preg_match("/".$pattern."/is",$cc)== 1){
    die();
} $bb="phpinfo()";
call_user_func($cc,$bb);
?>
```

3.设置`disable_functions`

```
$a=$_GET['db'];
$pattern =
"call_user_func|call_user_func_array|array_map|array_filter|ob_start|phpinfo|eval|asser
t|passthru|pcntl_exec|exec|system|escapeshellcmd|popen|chroot|scandir|chgrp|chown|shell
_exec|proc_open|proc_get_status|ob_start"; if(preg_match("/".$pattern."/is",$db)== 1){
die();
}
```

7.XSS

修复方案

1.HTTPONLY

2.`htmlspecialchars()`

```
<?php
if( array_key_exists( "name", $_GET ) && $_GET[ 'name' ] != NULL ) {
 checkToken( $_REQUEST[ 'user_token' ], $_SESSION['session_token' ], 'index.php' );
    $name = htmlspecialchars( $_GET[ 'name' ] );
      echo "<pre>Hello ${name}</pre>"; } ?>
```

使用`htmlspecialchars()`函数把预定义的字符进行HTML实体转义

3.`JavaScript编码`

8.CSRF

1.添加Referer字段

2.添加Token验证

3.添加验证码验证

4.利用Cookie安全策略:samesite属性

9.XXE

使用开发语言提供的禁用外部实体的方法

修复方案

PHP

```
libxml_disable_entity_loader(true);
```

Java

```
 DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();

 dbf.setExpandEntityReferences(false);
```

Python

```
from lxml import etree
xmlData = etree.parse(xmlSource,etree.XMLParser(resolve_entities=False))
```

Java