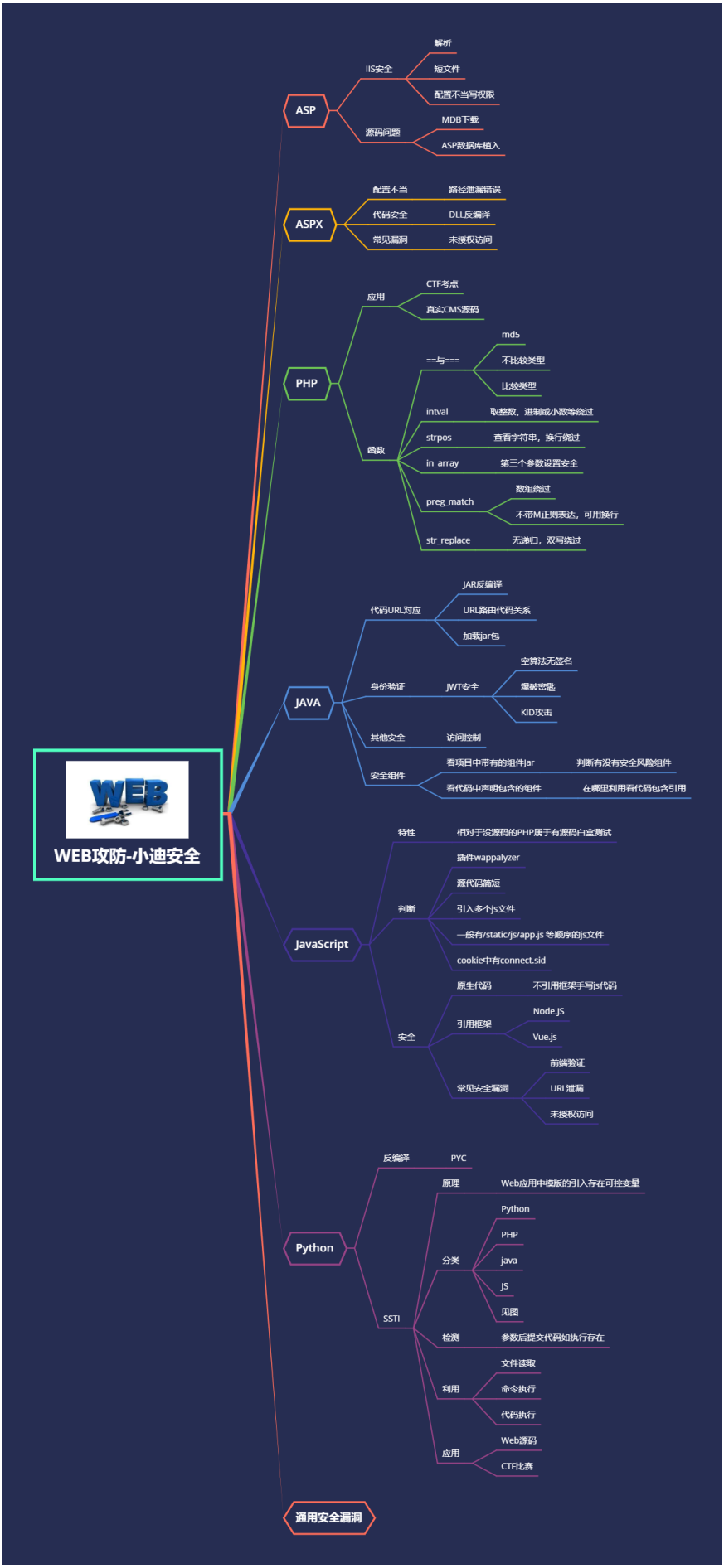
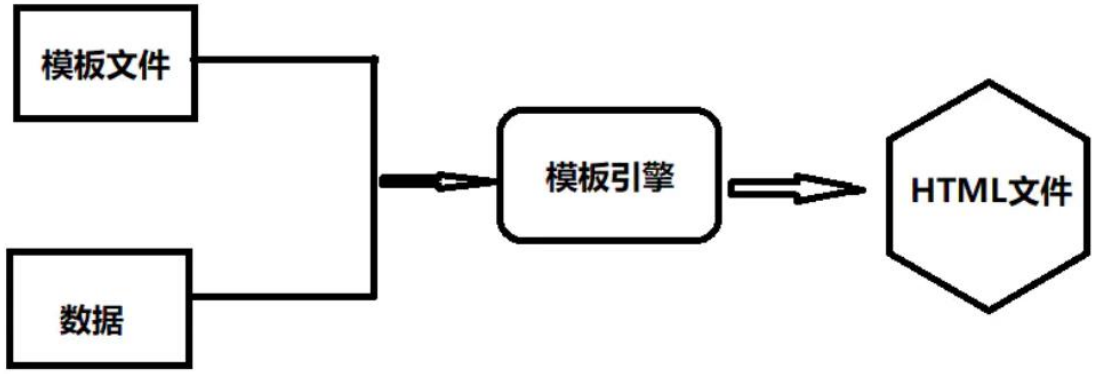
WEB 攻防-Python 考点&CTF 与 CMS-SSTI 模版注入&PYC 反编译


WEB攻防-小迪安全
ASP
IIS安全
解析
短文件
配置不当写权限
源码问题
MDB下载
ASP数据库植入
ASPX
配置不当
路径泄漏错误
代码安全
DLL反编译
常见漏洞
未授权访问
PHP
应用
CTF考点
真实CMS源码
函数
==与===
md5
不比较类型
比较类型
intval
取整数，进制或小数等绕过
strpos
查看字符串，换行绕过
in_array
第三个参数设置安全
preg_match
数组绕过
不带M正则表达，可用换行
str_replace
无递归，双写绕过
JAVA
代码URL对应
JAR反编译
URL路由代码关系
加载jar包
身份验证
JWT安全
空算法无签名
爆破密匙
KID攻击
其他安全
访问控制
安全组件
看项目中带有的组件jar
判断有没有安全风险组件
看代码中声明包含的组件
在哪里利用看代码包含引用
JavaScript
特性
相对于没源码的PHP属于有源码白盒测试
判断
插件wappalyzer
源代码简短
引入多个js文件
一般有/static/js/app.js 等顺序的js文件
cookie中有connect.sid
安全
原生代码
不引用框架手写js代码
引用框架
Node.JS
Vue.js
常见安全漏洞
前端验证
URL泄漏
未授权访问
Python
反编译
PYC
SSTI
原理
Web应用中模版的引入存在可控变量
分类
Python
PHP
java
JS
见图
检测
参数后提交代码如执行存在
利用
文件读取
命令执行
代码执行
应用
Web源码
CTF比赛
通用安全漏洞



Engine	Language	Burp	ZAP	tplmap	site done	known exploit	port	tags
jinja2	Python	✓	✓	✓	✓	✓	5000	{{%s}}
Mako	Python	✓	✓	✓	✓	✓	5001	${%s}
Tornado	Python	✓	✓	✓	✓	✓	5002	{{%s}}
Django	Python	✓	✓	×	✓	×	5003	{{ }}
(code eval)	Python	-	-	-	✓	-	5004	na
(code exec)	Python	-	-	-	✓	-	5005	na
Smarty	PHP	✓	✓	✓~	✓	✓	5020	{%s}
Smarty (secure mode)	PHP	✓	✓	✓~	✓	×	5021	{%s}
Twig	PHP	✓	✓	✓~	✓	×	5022	{{%s}}
(code eval)	PHP	-	-	-	✓	-	5023	na
FreeMarker	Java	✓	✓	✓	✓	✓	5051	<#%s > ${%s}
Velocity	Java	✓	✓	✓	✓	✓	5052	#set($x=1+1)${x}
Thymeleaf	Java	×	✓	x	✓	x	5053	
Groovy*	Java				×	×	×	×
jade	Java				×	×	×	×
jade	Nodejs	✓	✓	✓	✓	✓	5061	#{%s}
Nunjucks	JavaScript	✓	✓	✓	✓	✓	5062	{{%s}}
doT	JavaScript	×	✓	✓	✓	✓	5063	{{=%s}}
Marko	JavaScript				×	×	×	×
Dust	JavaScript	×	✓	✓~	✓	×	5065	{#%s}or{%s}or{@%s}
EJS	JavaScript	✓	✓	✓	✓	✓	5066	<%= %>
(code eval)	JavaScript	-	-	-	✓	-	5067	na
vuejs	JavaScript	✓	✓	✓~	✓	✓	5068	{{%s}}
Slim	Ruby	×	✓	x	✓	✓	5080	#{%s}
ERB	Ruby	✓	✓	✓	✓	✓	5081	<%=%s%>
(code eval)	Ruby	-	-	-	✓	-	5082	na
go	go	x	✓	x	✓		5090	na

#知识点：
1、PYC 文件反编译
2、Python-Web-SSTI
3、SSTI 模版注入利用分析

演示案例：

- PY 反编译-PYC 编译文件反编译源码
- SSTI 入门-原理&分类&检测&分析&利用
- SSTI 考点-CTF 靶场-[WesternCTF]shrine
- SSTI 考点-CMS 源码-MACCMS_V8.X 执行

#PY 反编译-PYC 编译文件反编译源码
pyc 文件是 py 文件编译后生成的字节码文件(byte code)，pyc 文件经过 python 解释器最终会生成机器码运行。因此 pyc 文件是可以跨平台部署的，类似 Java 的.class 文件，一般 py 文件改变后，都会重新生成 pyc 文件。
真题附件：http://pan.baidu.com/s/1jGpB8DS
反编译平台：
https://tool.lu/pyc/
http://tools.bugscaner.com/decompyle/
反编译工具：https://github.com/wibiti/uncompyle2

#SSTI 入门-原理&分类&检测&分析&利用
1、什么是 SSTI？有什么漏洞危害？
漏洞成因就是服务端接收了用户的恶意输入以后，未经任何处理就将其作为 Web 应用模板内容的一部分，模板引擎在进行目标编译渲染的过程中，执行了用户插入的可以破坏模板的语句，因而可能导致了敏感信息泄露、代码执行、GetShell 等问题。其影响范围主要取决于模版引擎的复杂性。
2、如何判断检测 SSTI 漏洞的存在？
-输入的数据会被浏览器利用当前脚本语言调用解析执行
3、SSTI 会产生在那些语言开发应用？
-见上图
4、SSTI 安全问题在生产环境那里产生？
-存在模版引用的地方，如 404 错误页面展示
-存在数据接收引用的地方，如模版解析获取参数数据

#SSTI 考点-CTF 靶场-[WesternCTF]shrine
1、源码分析 SSTI 考点
2、测试判断 SSTI 存在
3、分析代码过滤和 FLAG 存储
4、利用 Flask 两个函数利用获取
https://blog.csdn.net/houyanhua1/article/details/85470175

url_for()函数是用于构建操作指定函数的 URL
get_flashed_messages()函数是获取传递过来的数据
/shrine/{{url_for.__globals__}}
/shrine/{{url_for.__globals__['current_app'].config}}
/shrine/{{get_flashed_messages.__globals__}}
/shrine/{{get_flashed_messages.__globals__['current_app'].config}}

#SSTI 考点-CMS 源码-MACCMS_V8.X 执行
Payload:index.php?m=vod-search&wd={if-dddd:phpinfo()}{endif-dddd}
1、根据 wd 传递的代码找指向文件
2、index->vod->tpl->ifex->eval
3、构造 Payload 带入 ifex 并绕过后执行

涉及资源：
