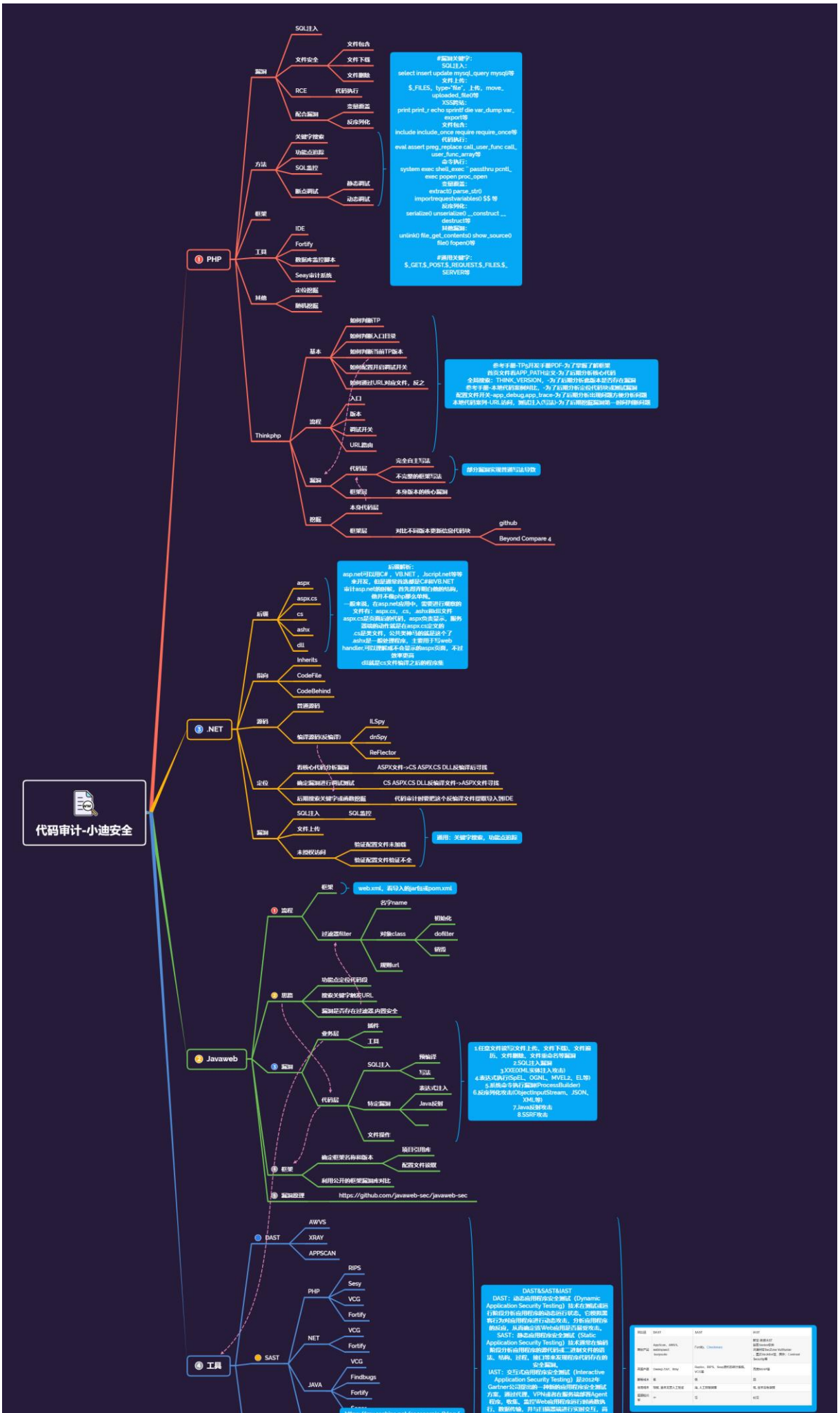
代码审计-Java 项目&SSTI&SSRF&XXE&XML&CNVD 模版&CTF 组件


代码审计-小迪安全
① PHP
漏洞
SQL注入
文件安全
文件包含
文件下载
文件删除
RCE
代码执行
配合漏洞
变量覆盖
反序列化
方法
关键字搜索
功能点追踪
SQL监控
断点调试
静态调试
动态调试
#漏洞关键字：
SQL注入：
select insert update mysql_query mysqli等
文件上传：
$_FILES、type="file"、上传、move_uploaded_file()等
XSS跨站：
print print_r echo sprintf die var_dump var_export等
文件包含：
include include_once require require_once等
代码执行：
eval assert preg_replace call_user_func call_user_func_array等
命令执行：
system exec shell_exec `` passthru pcntl_exec popen proc_open
变量覆盖：
extract() parse_str()
importrequestvariables() $$ 等
反序列化：
serialize() unserialize() __construct __destruct等
其他漏洞：
unlink() file_get_contents() show_source() file() fopen()等
#调用关键字：
$_GET,$_POST,$_REQUEST,$_FILES,$_SERVER等
框架
工具
IDE
Fortify
数据库监控脚本
Seay审计系统
其他
定位挖掘
随机挖掘
Thinkphp
基本
如何判断TP
如何判断入口目录
如何判断当前TP版本
如何配置开启调试开关
如何通过URL对应文件，反之
参考手册-TP5开发手册PDF-为了掌握了解框架
首页文件看APP_PATH定义-为了后期分析核心代码
全局搜索：THINK_VERSION，-为了后期分析版本是否存在漏洞
参考手册-本地代码案例对比，-为了后期分析定位代码快速测试漏洞
配置文件开关-app_debug,app_trace-为了后期分析出现问题方便分析问题
本地代码案例-URL访问，测试注入与写法-为了后期挖掘漏洞做一些铺垫和基础
流程
入口
版本
调试开关
URL路由
漏洞
代码层
完全自主写法
不完整的框架写法
部分漏洞实现思路写法导致
框架层
本身版本的核心漏洞
挖掘
本身代码层
框架层
对比不同版本更新信息代码块
github
Beyond Compare 4
③ .NET
后缀
aspx
aspx.cs
cs
ashx
dll
后缀解析：
asp.net可以用C#，VB.NET，Jscript.net等等来开发，但是通常首选都是C#和VB.NET
审计asp.net的时候，首先得弄明白他的结构，他并不像php那么单纯。
一般来说，在asp.net应用中，需要进行编译的文件有：aspx.cs，.cs，ashx和dll文件
aspx.cs是页面后台的代码，aspx负责显示，服务器端响应时就是在aspx.cs定义的
.cs是类文件，公共支持当的就是这个了
ashx是一般处理程序，主要用于写web handler.可以理解成不会显示的aspx页面，不过效率高点
dll就是cs文件编译之后的程序集
指向
Inherits
CodeFile
CodeBehind
源码
开源源码
编译源码(反编译)
ILSpy
dnSpy
ReFlector
定位
看核心代码分析漏洞
ASPX文件->CS ASPX.CS DLL反编译后寻找
确定漏洞进行调试测试
CS ASPX.CS DLL反编译文件->ASPX文件寻找
后期搜索关键字或函数挖掘
代码审计时需把这个反编译文件提取导入到IDE
漏洞
SQL注入
SQL监控
文件上传
未授权访问
验证配置文件未加载
验证配置文件验证不全
通用：关键字搜索，功能点追踪
④ Javaweb
框架
web.xml，看导入的jar包或pom.xml
① 流程
名字name
过滤器filter
对象class
初始化
dofilter
销毁
规则url
② 思路
功能点定位代码段
搜索关键字触发URL
漏洞是否存在过滤器内置安全
③ 漏洞
业务层
插件
工具
代码层
SQL注入
预编译
写法
特定漏洞
表达式注入
Java反射
文件操作
1.任意文件读写文件上传、文件下载、文件遍历、文件删除、文件重命名等漏洞
2.SQL注入漏洞
3.XXE(XML实体注入攻击)
4.表达式执行(SpEL、OGNL、MVEL2、EL等)
5.系统命令执行漏洞(ProcessBuilder)
6.反序列化攻击(ObjectInputStream、JSON、XML等)
7.Java反射攻击
8.SSRF攻击
④ 框架
确定框架名称和版本
项目引用库
配置文件读取
利用公开的框架漏洞库对比
⑤ 漏洞整理
https://github.com/javaweb-sec/javaweb-sec
⑤ 工具
DAST
AWVS
XRAY
APPSCAN
SAST
PHP
RIPS
Seay
VCG
Fortify
.NET
VCG
Fortify
JAVA
VCG
Findbugs
Fortify
DAST&SAST&IAST
DAST：动态应用程序安全测试（Dynamic Application Security Testing）技术在测试或运行阶段分析应用程序的动态运行状态，它模拟黑客行为对应用程序进行动态攻击，分析应用程序的反应，从而确定该Web应用是否易受攻击。
SAST：静态应用程序安全测试（Static Application Security Testing）技术通常在编码阶段分析应用程序的源代码或二进制文件的语法、结构、过程、接口等来发现程序代码存在的安全漏洞。
IAST：交互式应用程序安全测试（Interactive Application Security Testing）是2012年Gartner公司提出的一种新的应用程序安全测试方案。通过代理、VPN或者在服务端部署Agent程序，收集、监控Web应用程序运行时函数执行、数据传输，并与扫描器或进行实时交互，高