基于渗透测试的应用安全问题分析

步京蓬[1] 张奕然[2] 周 盛[1] 吕小兵[1] 杨 莹[1]
BU Jingpeng ZHANG Yiran ZHOU Sheng LYU Xiaobing YANG Ying

摘 要 针对当下网络安全形势提出通过渗透测试进行安全检测的方法，介绍了渗透测试概念及相关工作流程，结合部分检测实例对口令安全、Web 安全（文件上传漏洞、SQL 注入、XSS 攻击）、单点登录安全（明文信息标识、简单编码标识、固定加密凭据）、开源组件安全（Struts2 漏洞、Shiro 反序列化漏洞、Log4j 远程代码执行漏洞）等重点应用安全问题及渗透方法进行分析，提出问题解决措施，同时建议建立完善的渗透性安全检测机制，提升应用系统的安全防护水平，降低此类问题引发的安全风险。

关键词 渗透测试；应用安全；口令安全；Web 安全；单点登录安全；开源组件安全

doi：10.3969/j.issn.1672-9528.2024.08.039

0 引言

随着工业化与信息化的深度融合，网络和应用系统成为企业发展的重要辅助工具，网络安全的重要性日渐增加。目前企业网络安全防护通常依靠安全产品，通过防火墙进行网络访问控制[1]、物理隔离等手段抵御外部互联网攻击，对于应用系统自身存在的安全隐患则主要依靠漏洞扫描[2]、入侵检测[3]等进行被动审计分析，难以根据应用系统实际情况准确定位安全问题，缺乏在遭受攻击前主动发现并处理安全威胁的能力。渗透测试作为一种新兴的安全防护检测手段，“以攻代守”，可发现传统检测方法难以识别的攻击路径、安全脆弱点，打破了网络安全被动防护的局面。

1 渗透测试简介

1.1 相关概念

渗透测试[4]是一种安全风险检测、识别的方法，其模拟攻击者视角，在取得合法授权的情况下对信息系统进行主动攻击性尝试，以验证系统的安全性、保密性，及时发现相关脆弱点，促进系统维护人员针对性地采取防御措施，从而提高信息系统安全防护能力。

1.2 测试流程

渗透测试流程一般包括方案制定、信息收集[5]、漏洞探测、漏洞利用、权限提升、报告编写等环节，是通过收集目标的各类信息发现其安全缺陷，尝试利用其获取服务器权限，最终对测试问题进行分析、报告的安全检测过程。

1. 中航西安飞机工业集团股份有限公司 陕西西安 710089
2. 航空工业计算所 陕西西安 710068

2 应用安全问题

2.1 口令安全

在近年发生的网络安全事件中，口令安全[6]的比例正逐渐上升，因弱口令、口令复用等口令安全问题导致边界防护被突破的情况比比皆是。口令安全问题通常涉及应用系统、操作系统服务、Web 中间件、数据库等，可使用弱口令扫描工具进行探测，或根据单位、人员的相关信息生成针对性字典，通过 Hydra、Burpsuite 等工具进行暴力破解。

此类口令安全问题攻击成本较低，效果却十分明显，应在日常安全管理中做好相关排查工作，定期开展网络安全教育培训，增强人员口令安全意识。

2.2 Web 安全

开放式 Web 应用程序安全项目（OWASP）于 2023 年公布了全网 TOP 10 安全漏洞排名，其中传统 Web 安全[7]主要涉及访问控制、注入等漏洞，相比过去几年，该类型的安全问题比例已明显降低。尽管如此，传统 Web 安全问题带来的影响及危害仍不可忽视，尤其对于部分开发架构较为老旧的应用系统，依然存在以下高风险问题。

（1）文件上传漏洞

文件上传是应用系统常见的业务功能，如果该功能存在缺陷，未限制相关脚本文件格式，就可能产生文件上传漏洞，可通过冰蝎、蚁剑等工具连接上传的 WebShell 脚本文件，实现对网站甚至服务器的控制。该类漏洞操作难度较低，一旦利用成功则后果严重，因此安全风险较高。文件上传漏洞可采取服务端校验上传文件格式白名单的方法进行防护，防止 JS 校验、黑名单限制等方式被绕过。同时，应做好上传文件及路径信息的管理，对上传的文件重新命名，不暴露文件

名称和上传后存放的路径，并关闭路径的文件执行权限，使 WebShell 文件即使上传成功也无法连接运行。

（2）SQL 注入

应用系统在调用数据时通过查询与数据库产生交互，如果后端直接使用前端传入的参数执行查询 SQL 语句，就可能产生 SQL 注入漏洞。SQL 注入按照构造语句方式的不同分为联合注入、布尔注入、报错注入、时间注入等类型，可通过 SQLmap 工具针对注入点进行各类型的注入测试，获取数据库的名称、表信息、字段及数据等，属于危害较高的安全漏洞。

SQL 注入虽然存在多种不同的方式、类型，但漏洞产生的原因均为前端传入的可控查询参数未进行合规性判断、过滤，直接使用其内容拼接生成 SQL 语句，导致可构造参数进行注入、回显。针对 SQL 注入问题，可使用参数化的 SQL 查询方式，对查询参数的长度、类型进行限制，并过滤可能导致注入的相关字符，在 SQL 语句完成编译后再传入对应的参数值，避免参数中构造的 SQL 语句被执行。

（3）XSS 攻击

跨站脚本攻击简称为 XSS 攻击，是一种向 Web 系统页面插入脚本代码实施攻击的方式，主要分为存储型、反射型、DOM 型，通常存在于用户可控制输入的 Web 功能页面，如评论区、留言板、用户个人信息等。XSS 攻击的防护思路主要是控制不可信数据的输入，防止其插入页面中与正常代码一起被执行，通常采用标签过滤、解码规则规避等方法。

传统 Web 安全的问题机制各不相同，主要依靠分析具体原因、形成方法来针对性修复，从根源上解决问题。而 Web 应用系统业务覆盖广泛、页面数量较多，依靠漏洞扫描、入侵检测等安全设备难以全面覆盖且问题误报率高，无法在事前发现问题并完备处理，通过渗透测试技术对系统进行攻击性检测便可很好地化解这一窘境。

2.3 单点登录安全

大部分企业考虑应用系统管理及使用效率会建立门户系统，用户可通过门户系统进行统一身份认证后，以单点登录[8]方式认证登录其他应用系统，避免了使用不同系统时都需要进行身份认证的情况。门户系统的单点登录一般是通过其代理的接口请求各应用系统，在跳转登录时首先验证门户会话信息，以确认用户登录状态的合法性。当验证通过时将用户身份信息发送至单点系统，由单点系统确认后返回登录链接，自动跳转完成单点登录过程。会话信息是成功登录门户系统后产生的用户身份认证凭据，一般为了安全会采用 CA 认证[9]的方式，由 CA 服务器验证用户证书的合法性，认证通过后登录跳转至门户系统并形成本次会话。

在这一系列流程中，用户通过 CA 系统认证登录至门户系统，以及门户系统单点登录至其他系统时的认证过程是较安全的。首先，CA 系统的认证过程采用 HTTPS 协议，需要用户提供 CA 系统颁发的私有身份证书后才可进行口令验证操作，难以通过中间人攻击对其认证过程进行拦截、篡改。其次，单点登录过程中门户系统需对会话信息进行确认，通过后发送至单点应用的用户身份信息是在服务端交互的，客户端难以伪造会话对其产生影响。单点登录越权过程如图 1，其安全风险通常存在于后续应用认证过程，在单点应用通过门户系统确认用户身份后，返回的此次用户登录跳转链接及登录请求是可以在客户端被拦截、篡改的，于是可能存在以下问题。

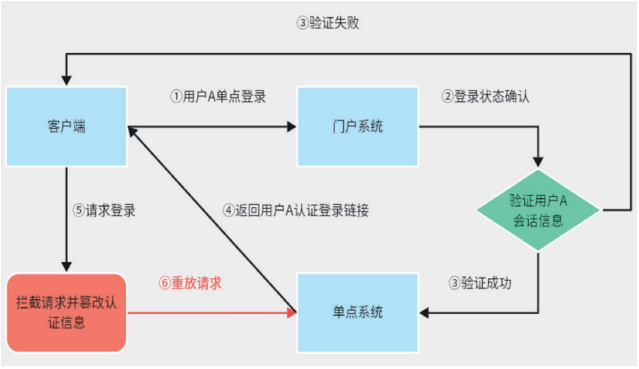

图 1 单点登录越权过程

（1）明文信息标识

GET 请求地址或 POST 请求数据中含有明文的用户身份信息，可通过 Burpsuite 工具拦截请求后替换该信息越权登录其他用户的系统。

（2）简单编码标识

GET 请求地址或 POST 请求数据中含有简单编码后的用户身份信息，如 BASE64 编码、URL 编码，将收集的其他用户身份信息采用相同方式编码，通过 Burpsuite 工具拦截请求并替换编码信息后越权登录他人系统。

（3）固定加密凭据

GET 请求地址或 POST 请求数据中含有固定的用户身份加密信息，即固定加密登录凭据，该凭据不会发生变化。可通过浏览器历史记录或网络嗅探方式获取该凭据，复用后越权登录至其他用户的系统。

（4）登录凭据泄露

有些应用系统的单点跳转请求是安全的，但由于其他业务接口的安全问题泄露了身份认证凭据，从而产生越权登录问题。如图 2 中应用系统原本是通过较为安全的临时身份认证凭据进行单点登录，却因其提供给门户系统的待办任务功能接口可通过明文用户身份信息请求任务表单，导致在响应数据中泄露了临时身份认证凭据。可通过 Burpsuite 工具请求接口获取其他用户的临时登录凭据，拦截并替换单点登录请

求中的凭据，即可成功越权登录其他用户的系统。

图 2 接口泄露凭据

门户系统单点登录应用广泛，由此引发的越权类问题造成了较大的安全风险隐患。在做好门户系统身份认证的同时，更应重点关注应用系统单点登录身份认证过程，及时采取临时加密凭据、随机 token、有效时间标记等安全措施，同时也要关注系统开放的业务功能接口可能导致的安全问题。

2.4 开源组件安全

应用系统通常使用开源组件满足多样化功能设计、运行实现需求，而软件供应链 [10] 也决定了存在开源组件的应用系统安全。目前企业应用供应源主要是国内具备开发资质的集成商、企业自主开发团队，在代码可控、数据可控方面一定程度上保证了安全，但对于引用的开源组件安全情况却容易忽视。以下列举了应用系统常见的高风险开源组件安全漏洞，并对其通过渗透测试进行验证的方法进行了简要分析。

（1）Struts2 漏洞

Struts2 是一个基于 MVC 设计模式的开发框架，在早期建设的应用系统中使用较多。其主要存在 S2-016、S2-045 等影响范围较大的漏洞，触发因素基本为通过漏洞执行构造的恶意 ognl 表达式来获取服务器相关权限。

S2-016 漏洞影响 Struts 2.3.15.1 之前的版本，形成原因是关于 Struts2 的请求未对 URL 中的 action、redirect 等字段进行处理，导致可通过构造自定义的 ognl 表达式执行系统命令获取服务器权限。S2-016 漏洞可利用 redirect 字段构造简单运算表达式（如加法运算），经 URL 编码拼接在请求 action 字段后，观察其响应情况，如返回的 location 字段中存在运算结果，则说明漏洞存在。

S2-045 漏洞影响 Struts 2.3.5-2.3.31、Struts 2.5-2.5.10 版本，是由于 Struts2 框架默认使用了 Jakarta 解析器处理文件上传请求，当 Content-Type 头出现解析错误时执行了错误信息中的 ognl 表达式，导致被利用获取服务器权限。S2-045 漏洞验证如图 3，可通过 Burpsuite 工具拦截问题页面的请求，对 Content-Type 头信息进行修改，构造 ognl 表达式来运算 5×6 并以 poc 字段在响应头信息中显示。可看到响应数据中返回了结果 30，证明漏洞存在。

图 3 S2-045 漏洞验证

（2）Shiro 反序列化漏洞

Shiro 是一个功能强大、易于集成的 Java 安全框架，通常用于 Web 应用程序的授权、身份认证、会话管理等。Shiro 框架存在反序列化漏洞（Shiro-550），影响 Shiro 1.2.4 及之前的版本，其主要成因是该版本范围的 Shiro 框架将 AES 密钥硬编码在源码中，而 Cookie 中的 rememberMe 字段会存储经序列化、AES 加密、base64 编码等过程处理后的用户信息，这就导致 AES 密钥泄露后可将恶意数据构造 rememberMe 字段值发送至服务端触发反序列化漏洞。

Shiro-550 漏洞验证如图 4，通过专用工具进行攻击性验证。首先，根据关键字判断目标页面返回包中的 rememberMe=deleteMe 字段，确认其使用了 Shiro 框架，将前期收集的默认 AES 密钥数据作为爆破字典；之后，通过工具批量发送 Cookie 中的 rememberMe 字段为 AES 密钥值的请求，以此爆破出当前 Shiro 框架使用的 AES 密钥；最后，尝试使用预置的构造链进行反序列化漏洞利用，成功获取服务器的命令执行权限，完成漏洞验证。

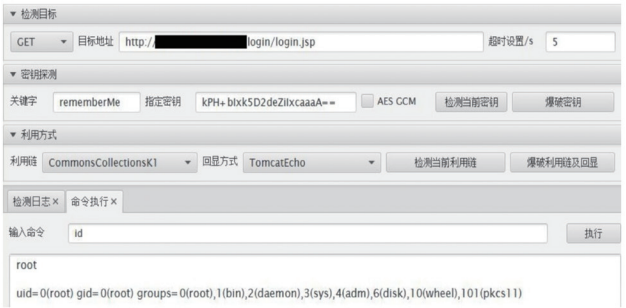

图 4 Shiro-550 漏洞验证

（3）Log4j2 远程代码执行漏洞

Log4j2 是一个基于 Java 的开源日志框架，其由于性能高、配置容易等优势，被广泛应用于各行业公司信息系统。2021 年底，Log4j2 被曝出存在远程代码执行漏洞，成功利用漏洞可执行任意代码来控制服务器。该漏洞影响 Log4j 2.x 至 2.15.0-rc1 版本，主要成因是 Log4j2 打印日志消息时调用 lookup 函数替换 ${} 格式的变量存在缺陷，

导致 JNDI 注入问题。JNDI 是一个可以通过服务名称进行对象调用的接口，通常用于访问 LDAP、RMI、DNS 等服务。此次注入问题就是通过 JNDI 调用 LDAP、RMI 服务实现的，攻击者可构造日志消息触发 Log4j2 的 lookup 函数进行解析，从而获取远程 LDAP、RMI 服务器的恶意代码加载到本地执行。

该漏洞可通过 JNDI 注入工具进行攻击验证，首先将反弹 shell 语句通过工具生成恶意 class 文件，并将包含该 class 文件的恶意 LDAP 服务地址构造为可触发 lookup 函数的字符串，例如：${jndi:ldap://192.168.41.131/hack}，之后将该字符串请求 Log4j2 接口进行处理。Log4j2 解析后请求地址为 192.168.41.131 的 LDAP 服务器，获取了 hack 路径下的 class 文件，随后在本机加载执行。结果如图 5，可看到监听端口已建立反弹 shell 的连接，完成漏洞验证。

图 5 Log4j2 远程代码执行漏洞验证

开源组件相关安全问题一般采取补丁升级或版本更新的方式解决，可部署 Web 应用防火墙针对特殊请求数据进行拦截、过滤。同时，应做好服务器的相关权限配置与状态监测，限制攻击者可获取利用的权限、资源，确保第一时间发现服务器状态、日志等异常，及时进行相关应急处置工作。

3 建立安全检测机制

为系统性排查、整改此类较为隐蔽的应用安全问题，以攻击者视角全面检测应用系统防护措施，应建立完善的应用系统渗透性安全检测机制，通过渗透检测不断发现该类问题引发的安全风险并及时妥善处置，从而进一步增强应用安全防护能力。

（1）全面检测

定期对应用系统开展渗透性安全检测，编写渗透测试方案明确待测应用系统范围、测试项、测试方法等。可考虑每半年对应用系统开展一次全面检测，并在新建系统正式使用前进行检测，保证检测范围全覆盖。

（2）问题整改

对测试中发现的问题进行总结分析，形成测试报告。同时，根据各项问题制定针对性整改方案，测试人员跟踪整改情况，在完成后及时进行复测，确保每项问题完全得到解决。

（3）安全规范

分析导致安全问题产生的因素，如版本未更新、配置错误、逻辑错误、编码缺陷等，将其融入应用系统建设与运维管理相关文件，形成避免类似问题发生的安全规范。

（4）形成制度

针对开展渗透性安全检测工作的相关要求、流程等形成一套专门的管理制度，以促进、指导该项工作的顺利落地。

4 结语

当前企业通过信息化手段进行业务融合的需求日益增加，如何全面保证信息系统的安全性已成为挑战。企业应用安全依靠传统被动防护手段已无法满足主动防御的安全需求，难以发现应用系统存在的安全隐患。建立完善的渗透性安全检测机制，以攻击者视角通过渗透测试技术检测应用系统安全问题，针对具体问题进行分析、整改，形成安全规范，防患于未然，才能更好地提升应用系统整体安全防护水平。

参考文献：

[1] 谢志奇 . 访问控制列表在网络安全及控制中的应用 [J]. 网络安全和信息化 , 2021(9):132-137.

[2] 蔡琴 . 基于网络安全的漏洞扫描模式的建立 [J]. 中国高新科技 ,2023(7):141-143.

[3] 王娜 , 狄秋燕 . 入侵检测技术在网络安全中的应用探讨 [J]. 网络空间安全 ,2023,14(5):89-93.

[4] 李维峰 .Web 应用程序渗透测试浅析 [J]. 网络安全技术与应用 , 2021(3):5-6.

[5] 巨腾飞 , 岳剑晖 .Web 渗透测试之信息收集研究 [J]. 网络安全技术与应用 ,2022(9):10.

[6] 雷丽 . 浅谈网络环境下的口令安全问题 [J]. 河北科技图苑 , 2008(1):39-41+62.

[7] 巢佳 .Web 安全问题及防护研究 [J]. 信息与电脑 (理论版), 2023,35(19):196-198.

[8] 耿云霄 , 许萌 , 蔡军霞 .Web 应用常见单点登录方式安全分析 [J]. 保密科学技术 ,2022(1):51-56.

[9] 吴庆敏 , 韩义勇 , 覃东海 , 等 .CA 认证系统的设计 [J]. 微型机与应用 ,2011,30(22):1-3+6.

[10] 冯兆文 . 从 Struts2 漏洞看办公内网软件供应链安全管理 [J]. 保密科学技术 ,2019(5):37-40.

【作者简介】

步京蓬（1996—），男，陕西西安人，本科，助理工程师，研究方向：网络和信息安全。

（收稿日期：2024-06-04）